大连理工大学

Dalian University of Technology

# 固定资产管理系统

—— 基于Java Platform,Standard Edition实现

姓名： 班级： 学号： 成绩：

姓名： 班级： 学号： 成绩：

Document-->>Index

# 欢迎来到固定资产管理系统

This is a simple document of our work. For more details, see as follows!<br>

<font color='green'>:-) &nbsp;</font> you can click items you want at your left side.

## **Outline**

\* `Requirement Analysis` - Something about what we done.

\* `Database Design` - For better usage and easy to code, database design is the basement.

\* `Classes ` - So many Classes we designed reference database.

\* `Project Structure & Project hierarchy` - We devided this homework into some divs for better use.

\* `The Usage of Generic Paradigm` - We benifited from generic paradigm so much.

\* `User Interface Show` - Show you the details of our homework.

\* `Experience` - Here you can see our expreience during developing this.

\* `About Us` - If you have something to say, see this page,please.

## **Project layout**

项目目录图：

FixedAssetsManageSystem: # Our Homework Overview

- src/

- dao/ Interfaces for CRUD

- AdminDao.java

- AssetsDao.java

- BaseDao.java

- UserDao.java

- dao.impl/ Implements of DAO

- AdminDaoImpl.java

- AssetsDaoImpl.java

- ClazzDaoImpl.java

- UserDaoImpl.java

- entity/ Entity for database table

- Admin.java

- Clazz.java

- FIxedAssets.java

- SubClass.java

- User.java

- todo.txt

- entrance/ The entrance of this software

- Main.java

- service/ For Business using

- AdminBusiness.java

- AssetsManager.java

- CatagoryBusiness.java

- UserBusiness.java

- util/ Utils for coding in some methods

- AssignmentTool.java

- DateFormat.java

- ConnectionFactory.java

- JDBCPool.java

- test/ My Code Test

- dao.impl.test/

- ···

- util.test/

- ···

- JRE System Library/ Java Runtime Environment

- Referenced Libraries/ Java Referenced Lbraries

- JUnit 4/ @Test using

- document/

- lib/ Necessary for Coding

- apache-commons-DbUtils.jar

- mysql-connector.java

## 需求分析：

# Requirement Analysis

This is the most important thing for us before developing. So let's have a look.

## References

We just using teacher's document as our reference. And we also search some words from the internet.

- <font color='red'>FixedAssets:</font> which value more than 2000￥。

- raw document:

随着公司业务不断发展，人员越来越多，固定资产的管理人员压力逐渐增大，需要建立数据库和使用软件系统来进行管理。

## Functions

- **管理员登录**:

要使用本系统必须先登录，并且只有固定资产管理员能够登录。登录的时候需要填写正确的用户名和密码，管理员可以修改自己的密码。系统允许有多个固定资产的管理员，每个管理员的权限相同。

- **固定资产信息管理**

- **固定资产类别管理**：

固定资产的类别在录入资产信息之前就定义好，录入资产信息之后可以增加新的类别。类别要求分为两级，如大类可以是办公外设、数码产品、计算机等；小类分为办公外设：传真机、复印机、打印机、其它；数码产品：数码相机、投影仪；计算机：笔记本电脑、台式机、服务器等。

类别应允许添加和删除操作。

- **固定资产信息管理**：

资产的信息包括编号、名称、类别、型号、价值、购买日期、状态、使用者、备注等。资产型号是文本信息，资产的状态分为：正常、维修、报废，使用者为空表示资产未被使用。资产应允许执行添加、修改和删除操作。

- 人员信息管理

这里的“人员”是指可能领用资产的人，人员的信息包括人员编号、姓名、职务和备注。

- 资产的领用

已经登记的人员，可以领用固定资产，领用时需要记录的信息有：设备编号、领用日期、管理员、用途、备注。只有状态为正常且使用者为空的资产才能被领用，在维修或者已经报废的资产不能够领用。

- 资产的归还

资产使用完毕后要归还给管理员，需要记录如下信息：设备编号，归还日期，管理员。

- 资产信息浏览和查询

- **按资产类别浏览**：

分类分成两级，要求能显示大类和小类下面的资产情况，资产状况。

- **查询资产信息**：

可以按照资产编号、资产类别、使用者查询详细信息。按照使用者查询显示该人员领用的所有资产

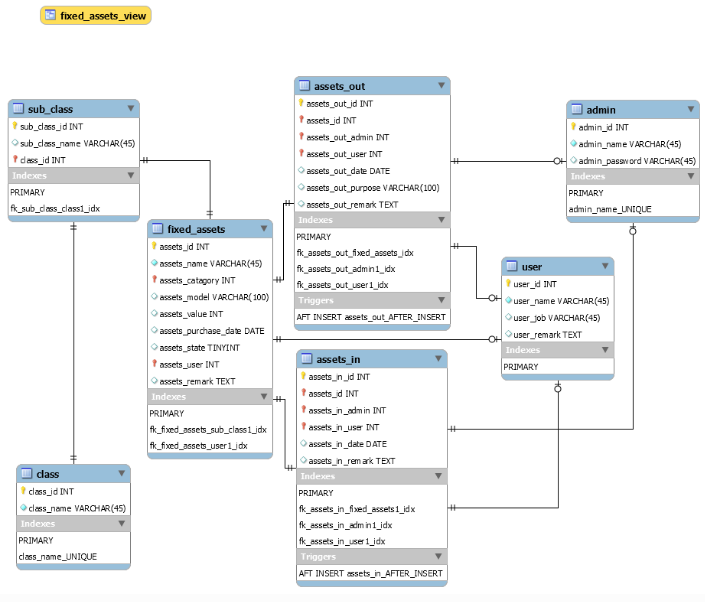
## 数据库设计：

# **Database Design**

我们添加了数据库连接池的配置，以及使用配置文件作为全局变量来实现软件的可移植性。基本上来说更换底层的数据库或者其他的模块，只需要手动的更改很少的代码就可以实现了。

This is also very important for the whole development. So we designed it reference our requirement analysis.

## Entity-Relationship View



## Database Table Property Details

- <font color='green' size='5'>资产大类信息表</font>`class`:

- `class\_id`: 大类的类编号

- `clas\_name`: 大类的类名称属性

- <font color='green'size='5'>资产小类信息表</font>`sub\_class`:

- `sub\_class\_id`: 小类的类编号

- `sub\_class\_name`: 小类的类名称属性

- `class\_id`: 作为外键起到约束作用的列，表示小类从属于哪个大类

- <font color='green'size='5'>资产单项信息表</font>`fixed\_assets`:

- `assets\_id`: 资产编号

- `assets\_name`: 资产名称

- `assets\_catagory`: 资产所属的小类的类型，因为知道了小类，其所属的大类就已经确定了

- `assets\_model`: 资产的型号

- `assets\_value`: 资产的价值或者价格

- `assets\_purchase\_date`: 资产的购买日期

- `assets\_state`: 资产的状态。0表示可借，1表示维修，2 表示报废。

- `assets\_user`: 已借出资产的持有人编号，未借出此项为空

- `assets\_remark`: 资产单项的备注信息

- <font color='green'size='5'>资产外借信息表</font>`assets\_out`:

- `assets\_out\_id`: 资产外借 编号信息

- `assets\_id`: 资产编号信息

- `assets\_out\_admin`: 处理资产外借的管理员

- `assets\_out\_user`: 资产外借时的使用人员的编号

- `assets\_out\_date`: 资产外借日期

- `assets\_out\_purpose`: 使用人员借用资产的目的，或者用途

- `assets\_out\_remark`: 资产外借的时候的备注信息

- <font color='green'size='5'>资产归还资产表</font>`assets\_in`:

- `assets\_in\_id`: 资产归还 编号信息

- `assets\_id`: 归还的资产的编号信息

- `assets\_in\_admin`: 处理归还资产的管理员的编号

- `assets\_in\_user`: 归还资产单项的使用人员的编号信息

- `assets\_in\_date`: 资产单项被归还的日期

- `assets\_in\_remark`: 资产单项被归还的备注信息

- <font color='green'size='5'>管理员信息表</font>`admin`:

- `admin\_id`: 管理员表的编号信息

- `admin\_name`: 管理员的姓名

- `admin\_password`: 管理员的密码信息

- <font color='green'size='5'> 使用人员信息表</font>`user`:

- `user\_id`: 使用人员的编号信息

- `user\_name`: 使用人员的名称

- `user\_job`: 使用人员的职位

- `user\_remark`: 使用人员本身的备注信息

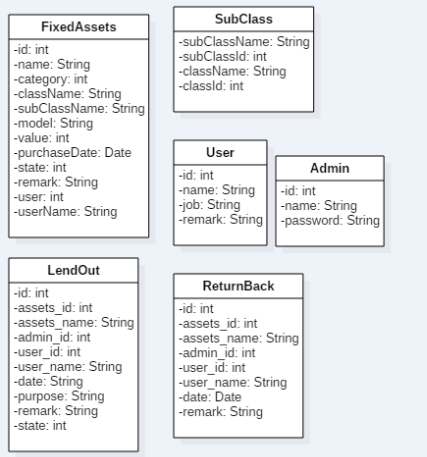
## 类图关系及设计:

# Classes View

UML can explain so many classes. So clear it is.

## Entity

Entities is created by referencing database tables.

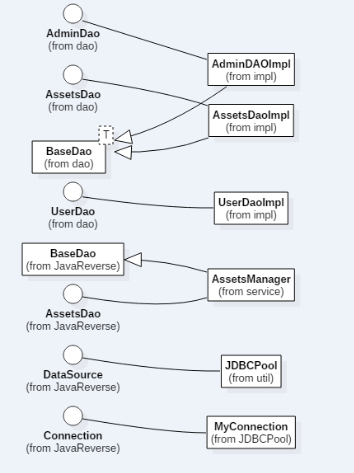
## Dao

Except basic opertions CRUD,there are so many interfaces in this package, which can depend on how we implments them.

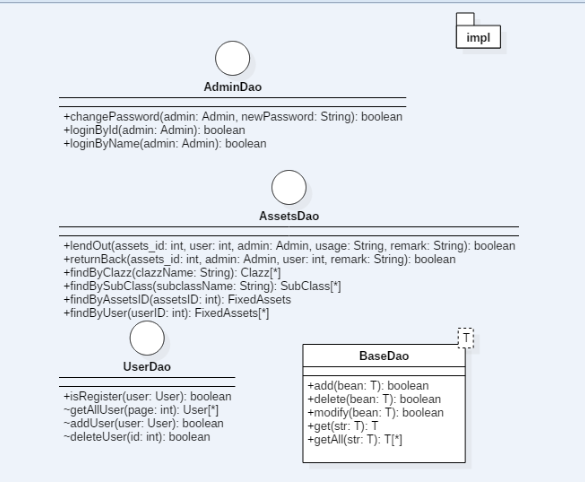
![Dao](http://a2.qpic.cn/psb?/V147oidM132mH5/XmY7IFB7iieH3HcpN3OYIjN.loebM9o6QnR7dRPEge4!/b/dOUAAAAAAAAA&bo=QALmAQAAAAADAIA!&rf=viewer\_4)

## DaoImpl

DaoImplments is the implements of dao interface. It's the burdge between Service and Database&entity。

## Service

Our Business is begin at here. Call those methods is very easy. It's mandatory to set them well.

## Type Hierachy

Hierachy is a good way to optimize our code. We can have a clear mind to solve those boring problems. And we can also read other's logic easily. It's very very important for our teamwork.Don't you think so?

![Type Hierachy](http://a2.qpic.cn/psb?/V147oidM132mH5/uMBokQKK14EvsX2Nu4oWgBLwL\*feRtWX\*cUx5bxSmqY!/b/dAwBAAAAAAAA&bo=fQEPAn0BDwIDACU!&rf=viewer\_4)

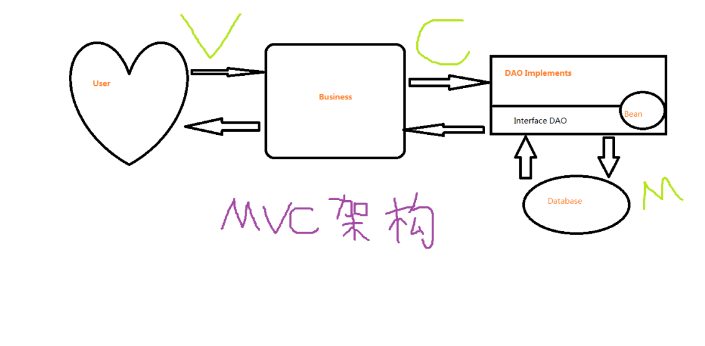
## 分层级架构设计：

# Project Structure And Project Hierachy

We all konw that the structure is very important. It's the core of one software.

## Project Hierachy

Like the picture as follows, it's the hierachy of our homework.



## Project Structure

The project directories is necessary for us to have a look.

FixedAssetsManageSystem: # Our Homework Overview

- src/

- dao/ Interfaces for CRUD

- AdminDao.java

- AssetsDao.java

- BaseDao.java

- UserDao.java

- dao.impl/ Implements of DAO

- AdminDaoImpl.java

- AssetsDaoImpl.java

- ClazzDaoImpl.java

- UserDaoImpl.java

- entity/ Entity for database table

- Admin.java

- Clazz.java

- FIxedAssets.java

- SubClass.java

- User.java

- todo.txt

- entrance/ The entrance of this software

- Main.java

- service/ For Business using

- AdminBusiness.java

- AssetsManager.java

- CatagoryBusiness.java

- UserBusiness.java

- util/ Utils for coding in some methods

- AssignmentTool.java

- DateFormat.java

- ConnectionFactory.java

- JDBCPool.java

- test/ My Code Test

- dao.impl.test/

- ···

- util.test/

- ···

- JRE System Library/ Java Runtime Environment

- Referenced Libraries/ Java Referenced Lbraries

- JUnit 4/ @Test using

- document/

- lib/ Necessary for Coding

- apache-commons-DbUtils.jar

- mysql-connector.java

### **Main structure**

![Main structure](http://a1.qpic.cn/psb?/V147oidM132mH5/J00H7eaWmDH94yJ84vn\*4u3myH6BhROHWFXughIIPHA!/b/dKsAAAAAAAAA&bo=1QHjAAAAAAADABI!&rf=viewer\_4)

### **Some jars and dependency**

![一些第三方的jar包以及相关的依赖项](http://a3.qpic.cn/psb?/V147oidM132mH5/XmI.UQieMn4C63TEGYP8yaonjLTHpbUTRObfu32GY4w!/b/dLAAAAAAAAAA&bo=FwKGAQAAAAADALc!&rf=viewer\_4)

### **The Details**

![上半部分](http://a1.qpic.cn/psb?/V147oidM132mH5/BwD3afUHB9egcSRKrJui0XY4x7WuS2ml6rVjqOzpMr8!/b/dKUAAAAAAAAA&bo=HAJSAgAAAAADAGs!&rf=viewer\_4)

![下半部分](http://a3.qpic.cn/psb?/V147oidM132mH5/fb0QnAA68i\*bQv.X\*Zw68J8iVLioUmU3RlU\*N2sxP8o!/b/dKoAAAAAAAAA&bo=IAIfAgAAAAADABo!&rf=viewer\_4)

## 泛型使用及心得：

# The Usage of Generic Paragidm

Generic Paradigm exists in java from JDK1.5 and later. It brings so many convenient for us coder.

## **sample 1**

package dao;

import java.util.List;

/\*\*

\* 用于增删改查信息的抽象基类

\*

\* @author 郭瑞彪

\*

\*/

public class BaseDao<T> {

/\*\*

\* 添加一条数据项

\*

\* @param bean

\*/

public boolean add(T bean) {

return false;

}

/\*\*

\* 删除一个数据项

\*

\* @param bean

\*/

public boolean delete(T bean) {

return false;

}

/\*\*

\* 修改一个数据项

\*

\* @param bean

\*/

public boolean modify(T bean) {

return false;

}

/\*\*

\* 根据查询条件获取一个数据项

\*

\* @param str

\* @return

\*/

public T get(T str) {

return null;

}

/\*\*

\* 根据查询条件获取所有的数据项

\*

\* @param str

\* @return

\*/

public List<T> getAll(T str) {

return null;

}

}



## **Sample 2**

```

/\*\*

\* 根据用户的用户编号来实现获取该用户名下借用的所有资产的列表

\*/

@Override

public List<FixedAssets> findByUser(int userID) {

try {

String sql = "select \* from fixed\_assets where assets\_user = '" + userID + "'";

Connection conn = ConnectionFactory.getConnection();

ResultSet rs = conn.createStatement().executeQuery(sql);

List<FixedAssets> ls = AssignmentTool.assignment2BeanList(rs);

return ls != null ? ls : null;

} catch (Exception e) {

throw new RuntimeException("根据用户编号来获取资产单项的详细信息的时候出错了！\n" + e);

}

}

```

## **Sample 3**

/\*\*

\* 罗列出所有的小类的信息

\*/

@Override

public List<SubClass> findBySubClass(String subclassName) {

try {

String sql = "select sub\_class\_name,class\_id from sub\_class where sub\_class\_name = '" + subclassName + "'";

Connection conn = ConnectionFactory.getConnection();

ResultSet rs = conn.createStatement().executeQuery(sql);

List<SubClass> ls = new ArrayList<SubClass>();

while (rs.next()) {

SubClass subclass = new SubClass();

subclass.setName(rs.getString("sub\_class\_name"));

int class\_id = rs.getInt("class\_id");

subclass.setClazzName(ClazzDaoImpl.getClassNameById(class\_id));

ls.add(subclass);

}

return ls != null ? ls : null;

} catch (Exception e) {

throw new RuntimeException("根据小类属性查找所有信息失败！" + e);

}

}

---

## **Sample 4**

/\*\*

\* 将从数据库中获得的结果集转换成一个装有返回结果的List集合

\*

\* @param rs

\* @return

\*/

public static List<FixedAssets> assignment2BeanList(ResultSet rs) {

try {

List<FixedAssets> lists = new ArrayList<FixedAssets>();

while (rs.next()) {

FixedAssets fas = new FixedAssets();

fas.setName(rs.getString("assets\_name"));

fas.setCatagory(rs.getInt("assets\_catagory"));

fas.setModel(rs.getString("assets\_model"));

fas.setValue(rs.getInt("assets\_value"));

fas.setPurchaseDate(rs.getDate("assets\_purchase\_date"));

fas.setState(rs.getInt("assets\_state"));

fas.setUser(rs.getInt("assets\_user"));

fas.setRemark(rs.getString("assets\_remark"));

lists.add(fas);

}

return lists;

} catch (Exception e) {

throw new RuntimeException("ResultSet to Bean Error！\n" + e);

}

}

## **Sample 5**

public static List<Object> result2BeanList(ResultSet rs, String classType) {

try {

List<Object> list = new ArrayList<Object>();

while (rs.next()) {

Object obj = null;

if (classType.equals("Clazz")) {

obj = new Clazz();

((Clazz) obj).setName(rs.getString("class\_name"));

} else if (classType.equals("SubClass")) {

obj = new SubClass();

((SubClass) obj).setClazzName(rs.getString("class\_name"));

((SubClass) obj).setName(rs.getString("sub\_class\_name"));

}

list.add(obj);

}

if (list != null) {

return list;

}

} catch (Exception e) {

throw new RuntimeException("结果集转换成BeanList时失败！\n" + e);

}

return null;

}

## **Sample 6**

/\*

\* 既然是一个数据库连接池，就必须有许多的连接，所以需要使用一个集合类保存这些连接 (non-Javadoc)

\*

\* @see javax.sql.CommonDataSource#getLogWriter()

\*/

private static LinkedList<Connection> list = new LinkedList<Connection>();

---

## 用户界面展示：

# User Interface Show

我们采取了分页机制以及面包屑导航，来给用户一个更加舒适的用户体验。具体我们可以从下面的截图中看出来。

We made the permission system carefully,such as.![Permisssion in Main](http://a2.qpic.cn/psb?/V147oidM132mH5/07oV4gaUpJu9G2bo2NMqYyjXDcFttBy6nCmlTm1l2FQ!/b/dHMAAAAAAAAA&bo=jQI\*AY0CPwEDACU!&rf=viewer\_4)

## 1、**Admin Login**

Only Admin can login this system for managing.![Admin Login](http://a2.qpic.cn/psb?/V147oidM132mH5/KzPZvVV\*O6x9Y9Ckzc3ARlYScxehIVLPP6kmgfHxPZg!/b/dF0BAAAAAAAA&bo=jQI\*AY0CPwEDACU!&rf=viewer\_4)

Change the Admin's Password.

![change admin password](http://a3.qpic.cn/psb?/V147oidM132mH5/A2msaxVnv8Y1RPuFi2z0ktUPo.KRK1UXc.7EVk26zJM!/b/dAcBAAAAAAAA&bo=7gJUAe4CVAEDCSw!&rf=viewer\_4)

## 2、**Fixed Assets Management**

Fixed Assets management is simple full of CRUD.

### a) **By Catagory**

class add:



**Class add result:**

![class\_add\_result](http://a3.qpic.cn/psb?/V147oidM132mH5/d\*D.KutbePj5xVDA.TIs0S5TBWb2lPd50gbNEzz8tok!/b/dAcBAAAAAAAA&bo=nQKNAZ0CjQEDCSw!&rf=viewer\_4)

**Class Delete:**

![Class Delete](http://a2.qpic.cn/psb?/V147oidM132mH5/hmSwr4k34Qbr0CydMxiqRgqViONFxSqFN0zZ17j9kAQ!/b/dAMBAAAAAAAA&bo=nQKNAZ0CjQEDACU!&rf=viewer\_4)

**Get all the classes:**

![get all classes](http://a3.qpic.cn/psb?/V147oidM132mH5/QK48iN\*40Y3PlI5RxEHsLOl8rOaahFCTLLajCI6kj2o!/b/dAcBAAAAAAAA&bo=jQI\*AY0CPwEDACU!&rf=viewer\_4)

![next page](http://a1.qpic.cn/psb?/V147oidM132mH5/d4jiQJqSRWgsJEh8EokytHrjMgQg5yUG20Rmb9RC58Y!/b/dAgBAAAAAAAA&bo=jQI\*AY0CPwEDACU!&rf=viewer\_4)

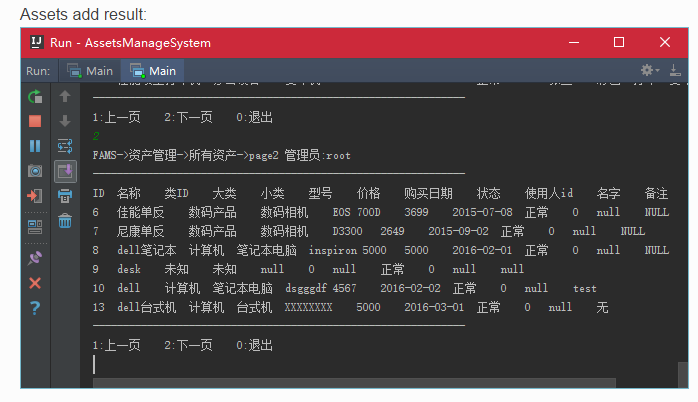
**Class management:**

![class Management](http://a3.qpic.cn/psb?/V147oidM132mH5/1pK10.7r\*iXzG1AwgyFw4DMTOB7fP7b8CpA7s3AQqC4!/b/dAcBAAAAAAAA&bo=jQI\*AY0CPwEDACU!&rf=viewer\_4)

### b) **By Assets Info**

**Assets Add:**

![Assets add](http://a2.qpic.cn/psb?/V147oidM132mH5/Z2kqRzrKsi9fIgVyRqXtyscmwtqCNGOWfdT8DyKcskA!/b/dAMBAAAAAAAA&bo=nQJqAZ0CagEDCSw!&rf=viewer\_4)



**Assets Delete:**

![Assets Delete](http://a1.qpic.cn/psb?/V147oidM132mH5/UkiGI\*XbRhsYkaj2QwpanUMG3OOw953WxlKmYiY62rc!/b/dAgBAAAAAAAA&bo=nQJpAZ0CaQEDACU!&rf=viewer\_4)

**Find Assets By Class:**

![find By class](http://a2.qpic.cn/psb?/V147oidM132mH5/sp7dG8tTdg.sqXuh0cma2bjBdb3HAhiR81YC7jER\*SM!/b/dAMBAAAAAAAA&bo=nQL8AZ0C\*AEDACU!&rf=viewer\_4)



## 3、User Info Management

The User management is controlled by admin.

**User Add:**

![User add](http://a2.qpic.cn/psb?/V147oidM132mH5/1KmFQWUqppDRN9\*t.sTV102uKAr.AJA4.M3lNDg6xrA!/b/dAMBAAAAAAAA&bo=3gLtAd4C7QEDCSw!&rf=viewer\_4)

**User Deleted:**

![UserDelete](http://a3.qpic.cn/psb?/V147oidM132mH5/vH2zMNmOYU3pwi4BeP3l2ZWDMi.o3rlm04HCfA.njZc!/b/dAcBAAAAAAAA&bo=3gLJAd4CyQEDACU!&rf=viewer\_4)

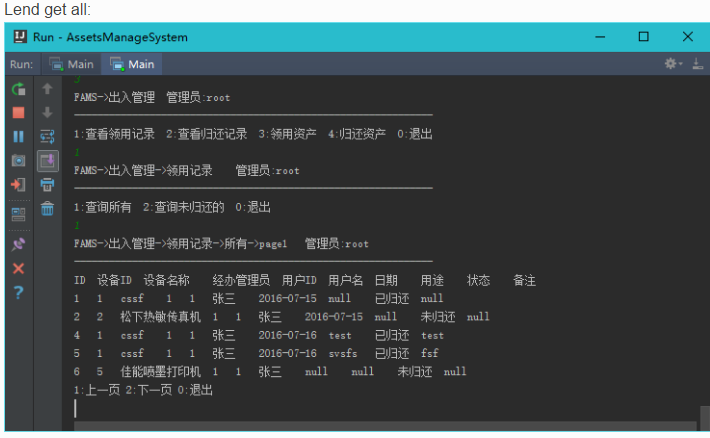
**Get All User list:**

![Get User list](http://a3.qpic.cn/psb?/V147oidM132mH5/f9Tqm63v7IkUMy.xbgtq.jD\*pZuEi4gUfJO\*u4K1oUY!/b/dAcBAAAAAAAA&bo=3gJyAd4CcgEDACU!&rf=viewer\_4)

**User Management:**

![Usermanagement](http://a2.qpic.cn/psb?/V147oidM132mH5/g31jrrJPK1aAi0T\*3vRMr1NaLwDKwv8CPGa4fkoVPKg!/b/dFoBAAAAAAAA&bo=3gJyAd4CcgEDACU!&rf=viewer\_4)

## 4、Assets Rent Out



**Lend Get not Return:**

![not return in lend](http://a3.qpic.cn/psb?/V147oidM132mH5/Nt0YPBNVPFNvqOJI6EXaLtOn1DcYPCXZcVn\*ONhUTho!/b/dAcBAAAAAAAA&bo=3gKqAd4CqgEDCSw!&rf=viewer\_4)

**Lend result log**

![lend log](http://a3.qpic.cn/psb?/V147oidM132mH5/0Zyxtrsb3WnKxF0Y2o.DtsV3zn3JGIrpPKSVeEE.igM!/b/dAcBAAAAAAAA&bo=7gJdAe4CXQEDACU!&rf=viewer\_4)

**Lend Out table：**

![lend out table](http://a3.qpic.cn/psb?/V147oidM132mH5/F7WCNRySHn6JQ6iNq1oaPECLOiRXZGVqoAV6BDIhBw4!/b/dAcBAAAAAAAA&bo=7gJdAe4CXQEDACU!&rf=viewer\_4)

## 5、Assets Return Back

Here you can see the return back info log.

![return back result](http://a2.qpic.cn/psb?/V147oidM132mH5/9m8eWa327b8De4r027v2QycmT3UZpvQCOXpkewR6UCM!/b/dAMBAAAAAAAA&bo=3gLXAN4C1wADACU!&rf=viewer\_4)

**Return get all:**

![return get all](http://a3.qpic.cn/psb?/V147oidM132mH5/OOFod34cMPd4yauVk6ilcW9ZWwplrf.xqMvZKWHW02g!/b/dAcBAAAAAAAA&bo=3gIzAd4CMwEDCSw!&rf=viewer\_4)



**Return Back Result:**

![return back result](http://a3.qpic.cn/psb?/V147oidM132mH5/0PHbh2TBDOSBT\*HwA90McRkO0YSVvNyGaddToMjg8wE!/b/dGgAAAAAAAAA&bo=7gJdAe4CXQEDACU!&rf=viewer\_4)

## 6、Assets List & Research

The core side function here.

**Find Assets By User ID:**

![by userid](http://a2.qpic.cn/psb?/V147oidM132mH5/ldkHDKhlZSJpAAHDQYTKxeweUihfhLE25U.YkT0U6po!/b/dGcAAAAAAAAA&bo=3gKlAd4CpQEDCSw!&rf=viewer\_4)

**Find Assets By User Name:**

![byusername](http://a3.qpic.cn/psb?/V147oidM132mH5/0DASSSN1sstx5DnCT8wWEBbrgKMDJt2ruW3ZLKio8Po!/b/dAcBAAAAAAAA&bo=3gKlAd4CpQEDACU!&rf=viewer\_4)

---

## 个人心得感悟：

# Our Experience

下面谈谈完成此次大作业的一些感悟吧。

## **郭瑞彪的感悟**

### **反思**

小学期刚开始的时候，觉得自己学了`Java`这么久了，自认为学的很好了。所以就没怎么在意课堂上相关知识点的讲解。而是学一些其他的比较新奇的东西，虽然在一定程度上获取到了更多的知识，但是却没能打下更加牢固的根基，现在想想这真的是有点“拣了芝麻丢了西瓜”的韵味。

其实仔细的思考一下，老马在课堂上所讲的基本上全是企业级或者开发级这个层面上常用的技术。相比于自己学习而言，很大程度上更加贴近于真实条件下软件的开发，所以是很有必要的。虽然自己学过这些所有的知识点，但还是会很有用的。

在此，我深刻的反思自己这种不认真的行为，确实是应该静下心来，稳扎稳打的学一波的。

### **收获**

说到收获，这就有很多了。

先从课堂上学到的一个小技巧吧。那是一个关于`JDBC`操作的小例子，具体见下面的代码。

/\*\*

\* 根据给定的bean 来实现资产单项的添加工作

\*/

@Override

public boolean add(Object bean) {

try {

String sql = "insert into fixed\_assets(assets\_name,assets\_catagory,assets\_model,assets\_value,assets\_purchase\_date,assets\_state,assets\_user,assets\_remark)"

+ " values(?,?,?,?,?,?,?,?)";

FixedAssets fas = (FixedAssets) bean;

Object[] params = { fas.getName(), fas.getCatagory(), fas.getModel(), fas.getValue(), fas.getPurchaseDate(),

fas.getState(), fas.getUser(), fas.getRemark() };

Connection conn = ConnectionFactory.getConnection();

PreparedStatement ps = conn.prepareStatement(sql);

for (int i = 0; i < params.length; i++) {

ps.setObject((Integer) (i + 1), params[i]);

}

int isSuccess = ps.executeUpdate();

conn.close();

ps.close();

return isSuccess == 1 ? true : false;

} catch (Exception e) {

throw new RuntimeException("固定资产插入数据时失败！\n" + e);

}

}

一个简单的小技巧，但是真的在开发中是那么的实用。我想这也是能经得起时间考验的程序员的智慧的结晶吧。

然后是关于团队开发的收获。

我们小组是两个人，我和姚晋宏。在开始编码之前，我们就关于关系型数据库的设计做了详细的讨论，然后根据项目需求制定了相关的表以及内部字段。这个过程让我也看到了自己这方面的不足。比如说本次表的设计，我的感悟是：

- 首先考虑独立于其他的表的那个表结构，这样的表最容易确定下来。

- 然后是考虑业务关系，可以这样的进行假设。当这个表中的数据发生改变的时候会引起其他的表的数据发生改变吗？ 如果会看看是一对一，多对一还是多对多的关系。然后设置外键约束即可。

- 有的时候需要添加视图`View`来保护数据库中的表，也有的时候会添加`Trigger` 或者`存储过程`来保证数据的联动效果。

- 最后根据表关系创建`E-R`关系图即可。

然后是团队项目合作过程的配合感悟，我和姚晋宏在开发的时候经过了详细的讨论，明确了各自的任务，前后端的开发互不干扰而且又能恰如其分的默契的配合，这样的团队开发才是最高效的团队开发。

### **方向**

开发的过程，真的是一个很锻炼人的过程。一直以为泛型，反射什么的技术没有什么实用性，现在看来还真的是`Too young to simple`啊。实际的开发中最常用而且也是最好用的估计就是这些东西了。

我的想法就是近期写出一套常用的工具集，然后不断的检验其正确定以及实用性。

## **姚晋宏的感悟**