【济南中心】JavaEE就业班同步笔记第一阶段： JavaWeb之核心...

小鲁哥哥 • 2017-2-19 18:19:25

【济南中心】JavaEE就业班同步笔记第一阶段：JavaWeb之核心技术--JDBC高级事务管理

1 使用MVC设计模式完成转账的案例:

1.1 需求:

设计一个页面,输入三个值,一个是付款人,一个是收款人,一个是转账的金额.不能出现付款人的钱被扣除而收款人没有收到钱的情况发生.而且要使用MVC的设计模式.

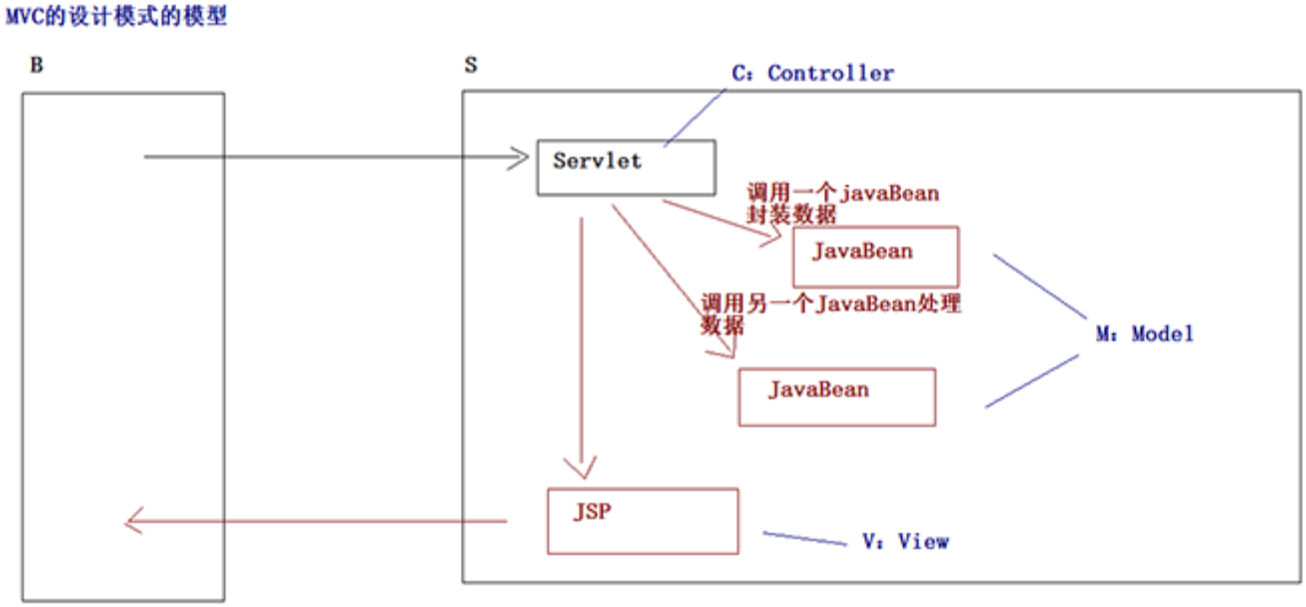
1.2 分析:

1.2.1 JSP的开发模式:

【动态网页开发模式的发展】

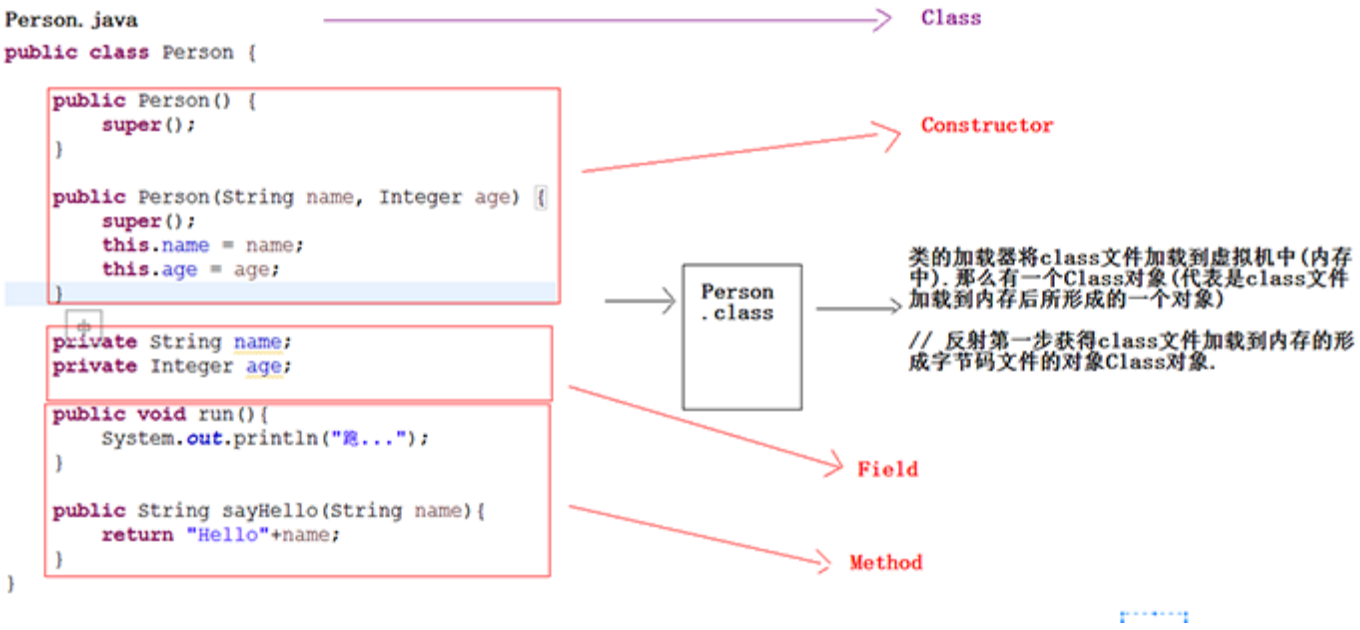
【JSP的开发模式一】： 了解

JSP + JavaBean
* 演示模式一的过程:
    * 在模式一开发中提供了一些JSP的标签:
<jsp:useBean> ,<jsp:setProperty >,<jsp:getProperty>
* 使用模式一进行简单的测试:
[mw_shl_code=html,true]<%
// 接收数据:
/* String username = request.getParameter("username");
String password = request.getParameter("password");
// 封装数据:
User user = new User();
user.setUsername(username);
user.setPassword(password); */
%>
<jsp:useBean id="user" class="com.itheima.demo1.domain.User"
scope="page"></jsp:useBean>
<%-- <jsp:setProperty property="username" name="user"/>
<jsp:setProperty property="password" name="user"/> --%>
<jsp:setProperty property="*" name="user"/>
<!-- 表单的元素的name属性的值与User中的属性名称一致 就可以自动封装 -->
<jsp:getProperty property="username" name="user"/>[/mw_shl_code]
【JSP的开发模式二】： 掌握
JSP + Servlet + JavaBean 称为MVC的设计模式.
MVC:
M:Model:模型层
V:View:视图层
C:Controller:控制层



【Java中的反射技术】（掌握）

反射:



【Java中的内省技术】(了解)

内省:用来获得JavaBean的属性及属性的get或set方法.

JavaBean:就是一个满足了特定格式的Java类:

* 需要提供无参数的构造方法:

* 属性私有
* 对私有的属性提供public的get/set方法.
内省的代码:
[mw_shl_code=java,true]public void demo1() throws Exception{
  // 获得了Bean的信息
  BeanInfo beanInfo = Introspector.getBeanInfo
(User.class);
  // 获得Bean的属性描述了
  PropertyDescriptor[] pds =
beanInfo.getPropertyDescriptors();
  for(PropertyDescriptor pd:pds){
   System.out.println(pd.getName());
   /*pd.getReadMethod(); // 获得get方法
   pd.getWriteMethod();// 获得set方法.
*/ }
}[/mw_shl_code]
使用内省封装一个MyBeanUtils:
[mw_shl_code=java,true]public class MyBeanUtils {
public static void populate(Object obj,Map<String,String[]>
map) throws Exception{
  // 获得类的所有的属性的名称:
  BeanInfo beanInfo = Introspector.getBeanInfo
(obj.getClass());
  // 获得类中所有的属性:
  PropertyDescriptor[] pds =
beanInfo.getPropertyDescriptors();
  for (PropertyDescriptor pd : pds) {
   if(map.containsKey(pd.getName())){
    Method method = pd.getWriteMethod();
    // 执行set方法:
    method.invoke(obj, map.get(pd.getName
())[0]);

```
  }
 }
}
}[/mw_shl_code]
```

【事务的概述】

什么是事务:

* 事务指的是逻辑上的一组操作,组成这组操作的各个逻辑单元要么一起成功,要么一起失败.

MYSQL的事务的管理： （了解）

* 创建一个账号的表:

```
[mw_shl_code=shell,true]create database web_13;
use web_13;
create table account(
id int primary key auto_increment,
name varchar(20),
money double
);
insert into account values (null,'张森',10000);
insert into account values (null,'凤姐',10000);
insert into account values (null,'如花',10000);[/mw_shl_code]
```

***** MYSQL的事务管理有两种方式:(MYSQL数据库事务默认是自动提交的.Oracle数据库事务默认是不自动提交.)

1.手动开启事务

* start transaction; -- 开启事务
* 多条sql;
* commit/rollback;

2.设置一个自动提交参数

* show variables like '%commit%'; -- 查看与commit相关参数.
* set autocommit = 0; -- 将autocommit参数设置为OFF.

Variable_name	Value
autocommit	OFF
innodb_commit_concurrency	0
innodb_flush_log_at_trx_commit	1

【JDBC中的事务管理】（掌握）
JDBC的事务的管理的API:
[mw_shl_code=shell,true]setAutoCommit(Boolean autoCommit)
commit()
rollback()[/mw_shl_code]
1.2.2 步骤分析:
【步骤一】：创建一个页面:
【步骤二】：导入JDBC相关的jar包和工具类.
【步骤三】：创建包结构.
【步骤四】：提交到Servlet-->Service-->DAO
【步骤五】：页面跳转:
1.3 代码实现:
1.3.1 准备工作:
1.3.2 代码实现:
1.3.3 DBUtils实现事务管理:
没有事务管理:
QueryRunner(DataSource ds)
Query(String sql,ResultSetHandler<T> rsh,Object... params)
Update(String sql,Object... params)
有事务管理:
QueryRunner()
Query(Connection conn,String sql,ResultSetHandler<T> rsh,Object... params)
Update(Connection conn,String sql,ResultSetHandler<T> rsh,Object params)
1.4 总结:
1.4.1 事务特性:
原子性：强调事务的不可分割.
一致性：强调的是事务的执行的前后，数据的完整性要保持一致.

隔离性：一个事务的执行不应该受到其他事务的干扰.
持久性：事务一旦结束(提交/回滚)数据就持久保持到了数据库.
1.4.2 如果不考虑事务的隔离性,引发一些安全性问题:
一类读问题:
* 脏读 :一个事务读到另一个事务还没有提交的数据.
* 不可重复读 :一个事务读到了另一个事务已经提交的update的数据,导致在当前的事务中多次查询结果不一致.
* 虚读/幻读 :一个事务读到另一个事务已经提交的insert的数据,导致在当前的事务中多次的查询结果不一致.
一类写问题:
* 引发两类丢失更新:
1.4.3 解决引发的读问题:
设置事务的隔离级别:
* read uncommitted :未提交读.脏读，不可重复读，虚读都可能发生.
* read committed :已提交读.避免脏读.但是不可重复读和虚读有可能发生.
* repeatable read :可重复读.避免脏读,不可重复读.但是虚读有可能发生.
* serializable :串行化的.避免脏读，不可重复读，虚读的发生.
***** MYSQL隔离级别：repeatable read Oracle隔离级别:read committed
1.4.4 演示脏读的发生:
分别开启两个窗口:A,B
分别查看两个窗口的隔离级别:select @@tx_isolation;
设置A窗口的隔离级别为:read uncommitted:
* set session transaction isolation level read uncommitted;
分别在两个窗口中开启事务:
* start transaction;
在B窗口完成转账的操作:
* update account set money = money - 1000 where name = '张森';
* update account set money = money + 1000 where name = '凤姐';

在A窗口查询数据:(钱已经到账---脏读)
* select * from account; -- A事务读到了B事务还没有提交的数据.
1.4.5 演示避免脏读，不可重复读发生
分别开启两个窗口:A,B
分别查看两个窗口的隔离级别:select @@tx_isolation;
设置A窗口的隔离级别为:read committed:
* set session transaction isolation level read committed;
分别在两个窗口中开启事务:
* start transaction;
在B窗口完成转账的操作:
* update account set money = money - 1000 where name = '张森';
* update account set money = money + 1000 where name = '凤姐';
在A窗口中进行查询:
* select * from account; -- 避免脏读.
在B窗口提交事务:
* commit;
在A窗口中再次查询:
* select * from account; -- 转账成功.(不可重复读:一个事务读到另一个事务中已经提交的update的数据,导致多次查询结果不一致.)
1.4.6 演示避免不可重复读:
分别开启两个窗口:A,B
分别查看两个窗口的隔离级别:select @@tx_isolation;
设置A窗口的隔离级别为:repeatable read:
* set session transaction isolation level repeatable read;
分别在两个窗口中开启事务:
* start transaction;
在B窗口完成转账的操作:
* update account set money = money - 1000 where name = '张森';
* update account set money = money + 1000 where name = '凤姐';
在A窗口查询:
* select * from account; -- 转账没有成功:避免脏读.
在B窗口提交事务:

* commit;
在A窗口中再次查询:
* select * from account; -- 转账没有成功:避免不可重复读.
1.4.7 演示避免虚读的发生:
分别开启两个窗口:A,B
分别查看两个窗口的隔离级别:select @@tx_isolation;
设置A窗口的隔离级别为:serializable:
* set session transaction isolation level serializable;
在A,B两个窗口中分别开启事务:
* start transaction;
在B窗口中完成一个insert操作:
* insert into account values (null,'王老师',10000);
在A创建中进行查询的操作:
* select * from account; -- 没有查询到任何结果.
在B窗口提交事务:
* commit; -- A窗口马上就会显示数据.

回帖

6条回帖

a690223483

虽然没学到 po主有点猛 保存先

沙发 • 2017-2-22 21:00:03

回帖

zhangkaitong

谢谢分享 赞赞赞！！！！！

藤椅 • 2017-2-24 13:32:27

回帖

wllpeter

6666666666666666666666666666666666666666

板凳 • 2017-4-19 22:07:02

回帖

wllpeter

感谢分享

报纸 • 2017-4-19 22:07:40

回帖

15369308090

力挺楼主，支持你 666

地板 • 2017-4-20 22:59:10

回帖

wuxuliang

找到这一篇，找不到下一篇，要是能整理一下就好了，好想来个打包下载。

7# • 2017-6-3 19:38:27

回帖