**DAO设计模式**

DAO的主要功能是数据操作，在程序标准开发架构中属于数据层的操作。标准开发架构如下：客户层《==》显示层《==》业务层《==》数据层《==》资源层

客户层：因为现在都采用B/S开发架构，所以一般客户都使用浏览器进行访问。

显示层：使用JSP/Servlet进行页面的效果显示。

业务层：会将多个原子性的DAO操作进行组合，组合成一个完整的业务逻辑。

数据层：（DAO）:提供多个原子性的DAO操作，如增加，修改，删除等，都属于原子性的操作。

在整个DAO中实际上是以接口为操作标准，即客户端依靠DAO实现的接口进行操作，而服务器端则要将接口具体实现。DAO由一下几个部分组成：

1. DatabaseConnection:专门负责数据库的打开与关闭操作的类。
2. VO：主要由属性，getter()，setter()方法组成，VO类中的属性与表中的字段相对应，每一个VO类的对象都表示表中的一条记录。
3. DAO：主要定义操作的接口，定义一系列数据库的原子性操作标准，如增加，修改，删除，查询等。
4. Impl：DAO接口的真是实现类，完成具体的数据库操作，但是不负责数据库的打开和关闭。
5. Proxy：代理实现类，主要完成数据库的打开和关闭，并且调用真实实现类对象的操作。
6. Factory:工厂类，通过工厂取得一个DAO的实例化对象。

**在真实的主题实现类中，根本就没有数据库打开和关闭的操作，只是通过构造方法取得了数据库的连接，真正负责打开和关闭是由代理类来完成的。**

DAO方法的命名：

1. 数据库更新的操作：doXxx,操作以do方式开头。
2. 数据库查询操作：findXxx,或则getXxx操作以find或get开头。
3. 在接口命名前面加一个I表示与类命名相区别。

在编写jsp代码时，所有的异常处理绝对不能使用out.println()进行页面输出，而需要将全部的异常交给后台输出，这样是为了避免安全隐患。

在一个jsp页面里绝对不允许导入sql包。一般来说只能导入“java.util.\*”包。

DAO设计模式，实现显示和数据相分离，减少jsp页面中的java代码。使程序更容易维护，更清晰。

**MVC设计模式**

MVC(模型-视图-控制器)：Model-View-Controller

**MVC开发都是以Servlet为主体展开的，Servlet接收所有客户端请求，然后根据请求调用相应的JavaBean，并将所有的显示结果交给JSP完成。**

MVC设计模式被分为以下三个核心层：

1. 显示层：主要负责接收Servlet传递的内容，并调用JavaBean，将内容显示给用户。
2. 控制层：主要负责所有的用户请求参数，判断请求参数是否合法，根据请求的类型调用Javabean执行操作并将最终处理的结果交由显示层进行显示。
3. 模型层：完成一个独立的业务操作组件，一般都是以JavaBean或则EJB的形式进行定义的。

在MVC设计模式中，最关键的部分是使用RequestDispatcher接口，因为内容都是通过此接口保存到jsp页面上进行显示的。

在MVC设计模式中，属性传递都是使用request属性范围传递。因为他的保存范围小，占用的内存也小。

**使用RequestDispatcher: RequestDispatcher rd = request.getRequestDispatcher(“要跳转的页面”); rd.forword(req,resp);**

在jsp页面中唯一允许导入的包只能是“java.util.\*”