**一、有关Java Web与MVC结构模式**

**学习过基本Java Web开发的人都已经了解了如何编写基本的Servlet，如何编写jsp及如何更新浏览器中显示的内容。但是我们之前自己编写的应用一般存在无条理性，对于一个小型的网站这样的编写没有任何问题，但是一但我们需要编写大型的web工程的话，我们现有的编写模式会造成web应用的可扩展性较差，而且一但出现问题不能准确的定位出问题出在哪里。**

**Java是一门应用设计模式比较广泛的语言。目前主流提出的23种设计模式均可在Java语言编写的程序中所应用。目前主流在Java Web应用中应用的最广泛的设计模式便是MVC模式，目前的主流Web框架大多也是基于MVC设计模式所编写的。**

**二、MVC在Java Web应用中的具体应用**

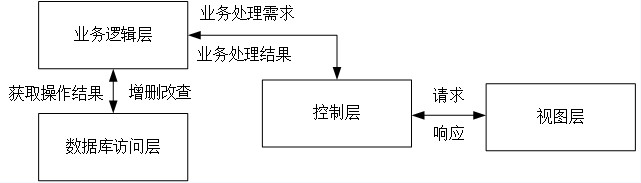
**所谓MVC，即Model-View-Controller。**

**（1）Model层：Model指模型部分，一般在应用中Model层包括业务处理层Service和数据访问层Dao。数据访问层主要是对数据库的一些操作的封装。业务处理层主要是用作将从Controller层获取的数据和数据库的数据进行桥接。除此以外，对复杂业务逻辑进行处理,比如事务处理。**

**（2）Controller层：Controller指控制部分，一般是对View层提交的请求为其设置对应的Servlet进行特定功能的处理，这里的进行特定功能的处理一般是编写在Model中的业务处理层中的。Controller一般只是在Web应用中充当一个中介者的作用。**

**（3）View层：View指视图部分，这一部分的内容是展示给用户实际进行交互的，通常使用JSP和HTML进行构建（个人比较喜欢以HTML嵌入JSP的方式来构建网页）。**

**综上来说，一个小型完整的基于MVC设计模式的Web应用程序的处理流程应该如下：**



**由上面的图中我们可以看出，用户在客户端（Web应用的客户端即为浏览器）中发出请求的时候，请求首先由View层的JSP/HTML将HTTP请求传给控制器中对应的Servlet，然后由Servlet负责调用Model层中的业务逻辑处理部分进行要求的处理，处理期间如果设计数据库的操作，则与数据库进行操作，最后全部操作结束之后，由业务逻辑层将结果发给控制层，控制层以HTTP响应的形式将结果发送回客户端。**

**(注意的几点:MVC模型中没有跨层操作,各层之间不应该有很强的耦合(面向接口编程),必须具有相对独立性,修改一个层的代码,不应该会影响到其他层代码跟着修改!)**

**三、一个MVC设计模式的小例子**

**我们的这个例子是运用MVC技术和基本的Java Web技术进行一个用户管理的Web应用的制作。**

**补充知识点:**





**java中dao层和service层的区别，为什么要用service？**

**首先解释面上意思，service是业务层，dao是数据访问层。**

**这个问题我曾经也有过，记得以前刚学编程的时候，都是在service里直接调用dao，service里面就new一个dao类对象，调用，其他有意义的事没做，也不明白有这个有什么用，参加工作久了以后就会知道，业务才是工作中的重中之重。**

**我们都知道，标准主流现在的编程方式都是采用MVC综合设计模式，MVC本身不属于设计模式的一种，它描述的是一种结构，最终目的达到解耦，解耦说的意思是你更改某一层代码，不会影响我其他层代码，表示层调用控制层，控制层调用业务层，业务层调用数据访问层。**

**初期也许都是new对象去调用下一层，比如你在业务层new一个DAO类的对象，调用DAO类方法访问数据库，这样写是不对的，因为在业务层中是不应该含有具体对象，最多只能有引用，如果有具体对象存在，就耦合了。**

**当那个对象不存在，我还要修改业务的代码，这不符合逻辑。好比主板上内存坏了，我换内存，没必要连主板一起换。我不用知道内存是哪家生产，不用知道多大容量，只要是内存都可以插上这个接口使用。这就是MVC的意义。**

**接下来说你感觉service的意义，其实因为你现在做东西分层次不是那么严格，在一个你们做东西业务本身也少，举个最简单的例子，你做一个分页的功能，数据1000条，你20条在一个页，你可以把这个功能写成工具类封装起来，然后在业务层里调用这个封装的方法，这才是业务里真正干得事，只要没访问数据库的，都要在业务里写。**

**比说你现在做一个用户模块：**

**1、假设现在你做这个功能会用到user表和权限表，那么你前台的页面访问action，action再去调用用户模块service，用户模块service判断你是操作user表还是权限表，如果你操作的是user表则service的实现类就去调用userDAO。如果是操作的是权限表则调用权限的DAO**

**2、也就是说DAO一定是和数据库的每张表一一对应，而service则不是。其实你一个项目一个service和一个DAO其实也一样可以操作数据库，只不过那要是表非常多，出问题了，那找起来多麻烦，而且太乱了**

**3、好处就是你的整个项目非常系统化，和数据库的表能一致，而且功能模块化，这样以后维护或者改错比较容易，性能也高一些**