**DO、PO、BO、DTO、VO等概念**

1. **POJO 概念:**

* 全称为: Plain Ordinary Java Object, 普通的Java 对象，一般用在数据层映射到数据库的类，类的属性与表字段一一对应

1. **PO 概念:**

* 全称为: Persistant Object, 持久化对象，与数据库结构映射的实体，数据库中的一条数据即为一个 PO 对象

1. **BO 概念:**

* 全称为: Business Object, 业务对象, 主要作用是把业务逻辑封装成一个对象，这个对象可以包括一个或多个其他对象，比如一个简历 BO 中有教育经历，工作经历社会关系等PO 对象

1. **DO 概念:**

* 全称为: Domain Object, 领域对象, 从现实世界中抽像出来的业务实体, 一般还包含 ORM 映射

1. **VO 概念:**

* 全称为: View Object, 主要对应页面展示的数据对象，一般继承自PO, 可以添加PO 中没有的字段,用来逻辑处理以及其他消息储存

1. **DTO 概念:**

* 全称为: Data Transfer Object, 数据传输对象,比如一张表有100个字段, 那么对应的PO就有100个属性(大多数情况下, DTO内部的数据结构来自多个表)但view 层只需要显示10个字段，没有必要把整个PO 对象传递到client，这时候我们就可以用只有这10个属性的DTO来传输给clinet，这样也不会表露server端表结构, 达到客户端后, 如果这个对象来对应页面显示, 它的身份就转为VO

1. **DAO 概念:**

* 全称为: Data Access Object, 数据访问对象, 一般所说的DAO层, 用于连接数据库与外层之间的桥梁，并持久化数据层对象

1. **JavaBean 概念:**

* JavaBean 是一种 Java 语言写成的可重用组建，它的规范必须符合特定的约定:
  + 这个类必须有一个公共的缺省构造函数
  + 这个类的属性用getter 和 setter 访问
  + 这个类可以被序列化

实用模型:

* 用户发出请求，表单的数据层被匹配为VO
* 展示层把 VO 转换为服务层对应的方法锁要求的DTO, 传输给服务层
* 服务层首先根据DTO 的数据构造一个 DO , 调用DO 的业务方法完成具体业务
* 服务层把 DO 转换位持久层对应的 PO , 调用持久层持久化方法, 把PO 传递给它完成持久化操作