Java中的各种bean对应的意义(VO,PO,BO,QO, DAO,POJO,DTO)

**VO(value object) 值对象**

通常用于业务层之间的数据传递，用 new 关键字创建，由 GC 回收的，和 PO 一样也是仅仅包含数据而已。但应是抽象出的业务对象 , 可以和表对应 , 也可以不 , 这根据业务的需要 . 个人觉得同 DTO( 数据传输对象 )一样 在 web 上传递。

**PO(persistant object)  持久对象**

在 o/r 映射的时候出现的概念，如果没有 o/r 映射，没有这个概念存在了。通常对应数据模型 ( 数据库 ), 本身还有部分业务逻辑的处理。可以看成是与数据库中的表相映射的 java 对象。最简单的 PO 就是对应数据库中某个表中的一条记录，多个记录可以用 PO 的集合。 PO 中应该不包含任何对数据库的操作。 

**TO(Transfer Object)  数据传输对象**

在应用程序不同 tie( 关系 ) 之间传输的对象，用于数据交互

**BO(business object) 业务对象**

从业务模型的角度看 , 见 UML 元件领域模型中的领域对象。封装业务逻辑的 java 对象 , 通过调用 DAO 方法 , 结合 PO,VO 进行业务操作。 主要作用是把业务逻辑封装为一个对象。这个对象可以包括一个或多个其它的对象。 比如一个简历，有教育经历、工作经历、社会关系等等。 我们可以把教育经历对应一个 PO ，工作经历对应一个 PO ，社会关系对应一个 PO 。 建立一个对应简历的 BO 对象处理简历，每个 BO 包含这些 PO 。 这样处理业务逻辑时，我们就可以针对 BO 去处理。

**QO(Query object)  查询对象**

一般用于组装查询参数

**POJO(plain ordinary java object) 简单无规则 java 对象**

若一些bean类觉得简单，或者仅仅作为中间件，那么可以归类于pojo

**DAO(data access object)  数据访问对象**

是一个 sun 的一个标准 j2ee 设计模式， 这个模式中有个接口就是 DAO ，它负持久层的操作。为业务层提供接口。此对象用于访问数据库。通常和 PO 结合使用， DAO 中包含了各种数据库的操作方法。通过它的方法 , 结合 PO 对数据库进行相关的操作。夹在业务逻辑与数据库资源中间。配合 VO, 提供数据库的 CRUD 操作 ...

**DTO(Data Transfer Object) 数据传输对象**

主要用于远程调用等需要大量传输对象的地方。 比如我们一张表有 100 个字段，那么对应的 PO 就有 100 个属性。 但是我们界面上只要显示 10 个字段， 客户端用 WEB service 来获取数据，没有必要把整个 PO 对象传递到客户端， 这时我们就可以用只有这 10 个属性的 DTO 来传递结果到客户端，这样也不会暴露服务端表结构 . 到达客户端以后，如果用这个对象来对应界面显示，那此时它的身份就转为 VO