设计 web 服务器

1 PO 对象的创建

PO(persistant object) 持久对象

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。

User.java

```
public class User implements Serializable{
    private Integer user_id;
    private String username;
    private String userpass;
    private String email;
}
```

Item.java

```
public class Item implements Serializable {
    // 标识属性
    private Integer item_id;
    // 物品名称
    private String item_name;
    // 物品 Remark
    private String item_remark;
  // 物品描述
    private String item_desc;
    // 物品添加时间
    private Date addtime;
    // 物品结束拍卖时间
    private Date endtime;
```

```
    // 物品的起拍价
    private double init_price;
    // 物品的最高价
    private double max_price;
    // 该物品的所有者
    private User owner;
    // 该物品所属的种类
    private Kind kind;
    // 该物品的赢取者
    private User winer;
    // 该物品所处的状态
    private State itemState;
    // 该物品对应的全部竞价记录
    private Set<Bid> bids = new HashSet<Bid>();
}
```

Kind.java

```
public class Kind implements Serializable{
    // 标识属性
    private Integer kind_id;
    // 种类名
    private String kind_name;
    // 种类描述
    private String kind_desc;
    // 该种类下的所有物品
    private Set<Item> items = new HashSet<Item>();
}
```

State.java

```
public class State implements Serializable{
    // 标识属性
```

```
    private Integer state_id;
    // 物品的状态名
    private String state_name;
    // 该状态下的所有物品
    private Set<Item> items = new HashSet<Item>();
}
```

Bid.java

```
public class Bid implements Serializable{
    // 标识属性
    private Integer bid_id;
    // 竞价的价格
    private double bid_price;
    // 竞价的日期
    private Date bid_date;
    // 本次竞价所竞拍的物品
    private Item item_id;
    // 参与竞价的用户
    private User user_id;}
```

2 VO 对象的创建

VO(value object) 值对象

通常用于业务层之间的数据传递，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),在 web 上传递。

ItemBean.java

```
package com.xbmu.business;
import java.util.Date;
public class ItemBean{
    private Integer id;
    private String name;
```

```
    private String desc;
    private String remark;
    private String kind;
    private String owner;
    private String winer;
    private String state;
    private double initPrice;
    private double maxPrice;
    private Date addTime;
    private Date endTime;

    // 无参数的构造器
    public ItemBean()
    {
    }
    // 初始化全部属性的构造器
    public ItemBean(Integer id , String name , String desc , String r
emark,String kind , String owner , String winer , String state ,doubl
e initPrice , double maxPrice , Date addTime , Date endTime)
    {
        this.id = id;
        this.name = name;
        this.desc = desc;
        this.remark = remark;
        this.kind = kind;
        this.owner = owner;
        this.winer = winer;
        this.state = state;
        this.initPrice = initPrice;
```

```
        this.maxPrice = maxPrice;
        this.addTime = addTime;
        this.endTime = endTime;
    }
//...省略 set 和 get 方法
}
```

BidBean.java

```
package com.xbmu.business;
import java.io.Serializable;
import java.util.Date;
public class BidBean implements Serializable
{
    private int id;
    private String user;
    private String item;
    private double price;
    private Date bidDate;
    // 无参数的构造器
    public BidBean()
    {
    }
    // 初始化全部属性的构造器
    public BidBean(int id , String user , String item ,
        double price , Date bidDate)
    {
        this.id = id;
        this.user = user;
        this.item = item;
        this.price = price;
```

```
        this.bidDate = bidDate;
    }
//...
}
```

KindBean.java

```
package com.xbmu.business;
public class KindBean{
    private int id;
    private String kindName;
    private String kindDesc;
    // 无参数的构造器
    public KindBean()
    {
    }
    // 初始化全部属性的构造器
    public KindBean(int id , String kindName , String kindDesc)
    {
        this.id = id;
        this.kindName = kindName;
        this.kindDesc = kindDesc;
    }
```