# 任务系统

1.线程安全问题

基于玩家标识与线程，单玩家操作，由同一个线程处理，不会有线程安全问题

* 缺实现一个线程池，此线程池可用于绑定标识线程，以标识决定执行线程
* 实现思路：使用concurrenthashmap,AotimicLong自增，fix固定线程池，实现runnable接口的，带有标识方法的task类实现

2.内存问题

粒度基于玩家，以玩家任务为缓存加载粒度，玩家上线加载，下线一段时间后清理，防止频繁登录掉线

* 缺实现缓存
* 使用guava cache作为缓存组件，实现登出监听类，登出方法中携带缓存句柄，在此做间隔时间清除缓存处理

3.代码耦合问题

利用事件解决，从基础功能，触发任务时间，产生相应任务数据，与基础功能解耦

* 事件同步异步问题若用同步则不会用问题，**若用异步，任务模板里，判断是否完成等操作，需要依赖其它数据需要做额外处理**

4.扩展性问题

以任务模板的形式实现，若有新的任务类型，则增加任务模板

触发方式也可扩展，例如基于某任务完成才触发任务，或基于某个特殊条件

策划可配置

## 扩展问题回答

任务系统里面，用了模板模式，事件模式解耦（具体用了什么模式不清楚，反正扩展性没问题，符合开闭原则添加任务模板，数据生产消费解耦）

# 背包系统（公会仓库，玩家主背包）

1.线程安全问题

**在一个完整业务操作内（内部）**，基于玩家标识与线程，单玩家操作，由同一个线程处理，不会有线程安全问题

2.内存问题

玩家主背包粒度基于玩家，以玩家背包为缓存加载粒度，玩家上线加载，下线一段时间后清理，防止频繁登录掉线

公会仓库粒度基于公会，以公会为缓存加载粒度，游戏开服加载，关服清理

1. 代码耦合问题

复用背包结构，不管是玩家主背包还是公会仓库结构其实一样

复用操作抽出背包操作的增删查改，操作流程可复用，使用模板模式抽出公用部分，差异部分在具体子类实现

操作分两种：

内部操作核心业务逻辑和缓存代码挂在在domain

外部操作，再内部操作封装一层，可灵活处理，枷锁等处理线程安全问题

## 扩展问题回答

1两个公会成员，同时取同一公会仓库物品，线程安全？

首先是相关操作，公会成员自身背包是否已满，若否，则可提取公会物品，更新玩家背包

具体做法：

请查看文档《公会仓库.PDF》

2 玩家发现整理背包操作，又迅速发起消耗操作，线程安全？

独立的玩家标识线程池，对玩家操作交个该线程池,不会有线程安全问题

# 交易系统

## 扩展问题回答

1 面对面交易流程？线程安全？

具体做法：

请查看文档《面对面交易.PDF》

# 账号系统

1 两个客户端同一时间相互顶号，有什么隐患？

不同连接，同一账号（**好像不会有啥问题。。。不清楚**）

# BOSS系统

1.boss额外奖励最后一击玩家，线程安全，重复奖励？

创建独立的boss标识线程池，对boss操作交个该线程池,不会有线程安全问题

发奖励时交个玩家标识线程池，不会有重复奖励问题