rBook详细设计规约

1652668 张嘉洛

1652669 俞杨

1652671 杜鹏

1652678 国明涵

修订历史：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编写日期 | 版本 | 说明 | SPEG | 修订人 |
| 2018.12.26 | 1 | 初始编写 | 软件工程课程杜庆峰老师 | 俞杨 |

## 引言

* 1. 编写目的

本规约系在概要设计规约基础上对类设计进一步的细化，明确其属性、方法及协作关系，以作为代码生成的依据。

* 1. 编写依据

[rBook概要设计规约](rBook概要设计规约.docx)

* 1. 参考资料

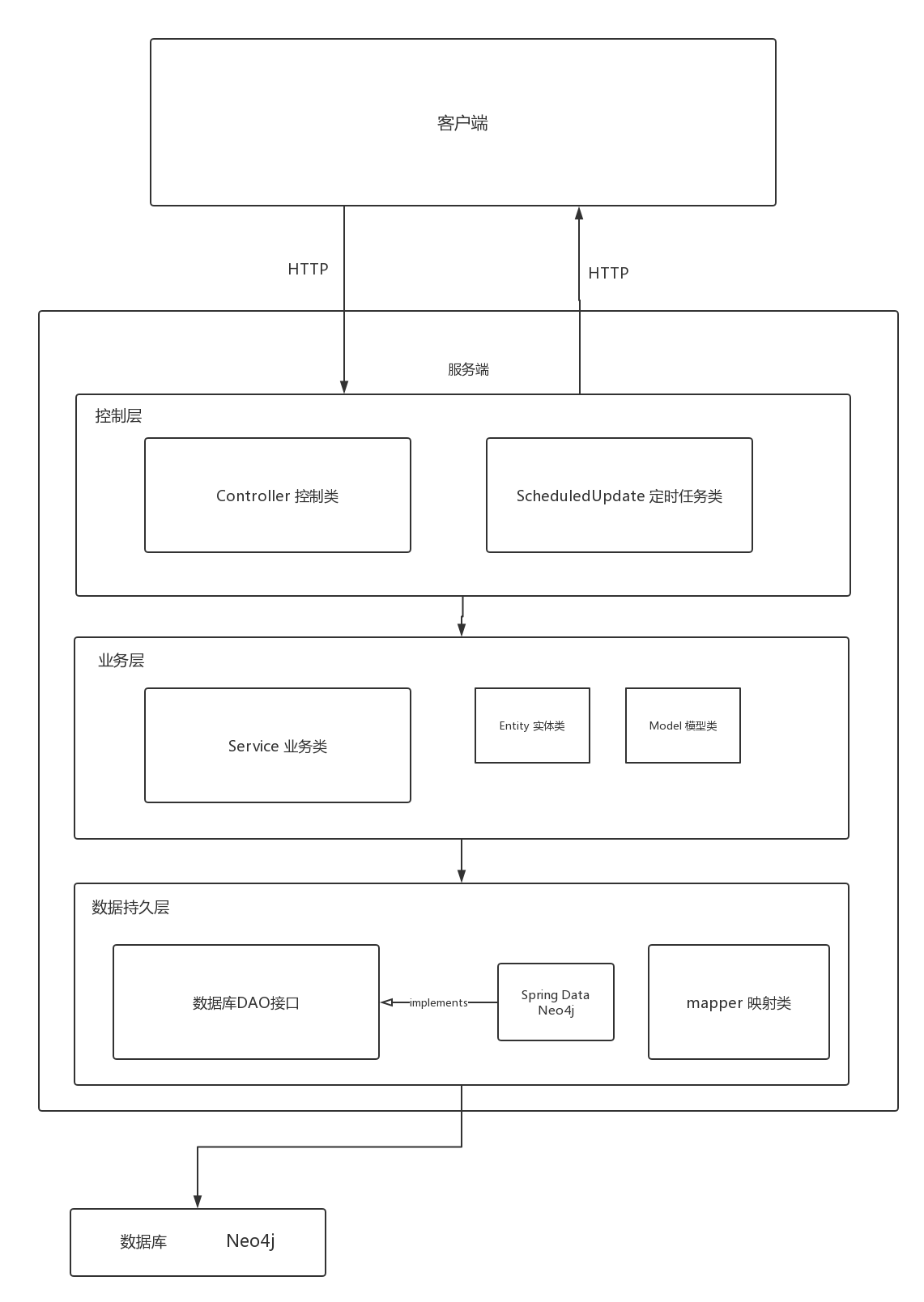
[Spring Boot Reference Document](https://docs.spring.io/spring-boot/docs/2.1.1.RELEASE/reference/htmlsingle/)

[Spring Data Neo4j Reference Document](https://docs.spring.io/spring-data/neo4j/docs/5.1.3.RELEASE/reference/html/)

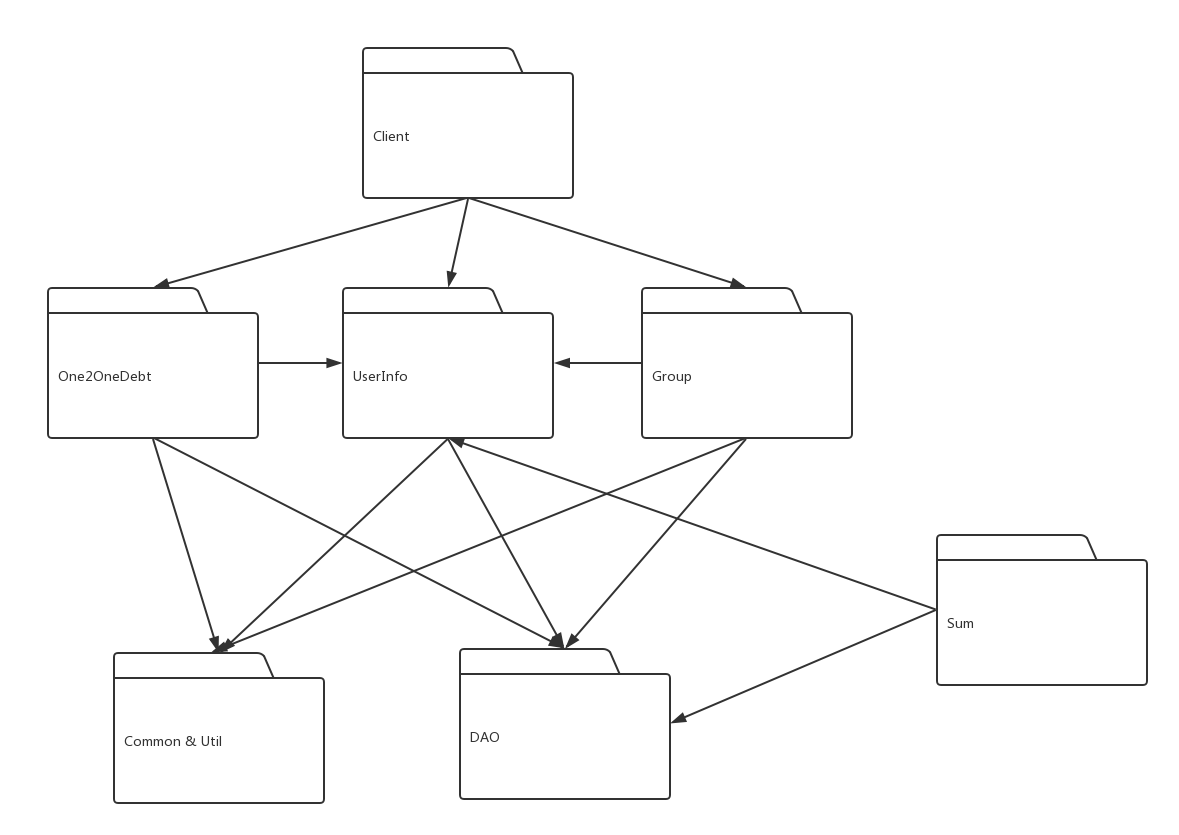
## 系统软件结构

同概要设计规约

系统层次架构如下：

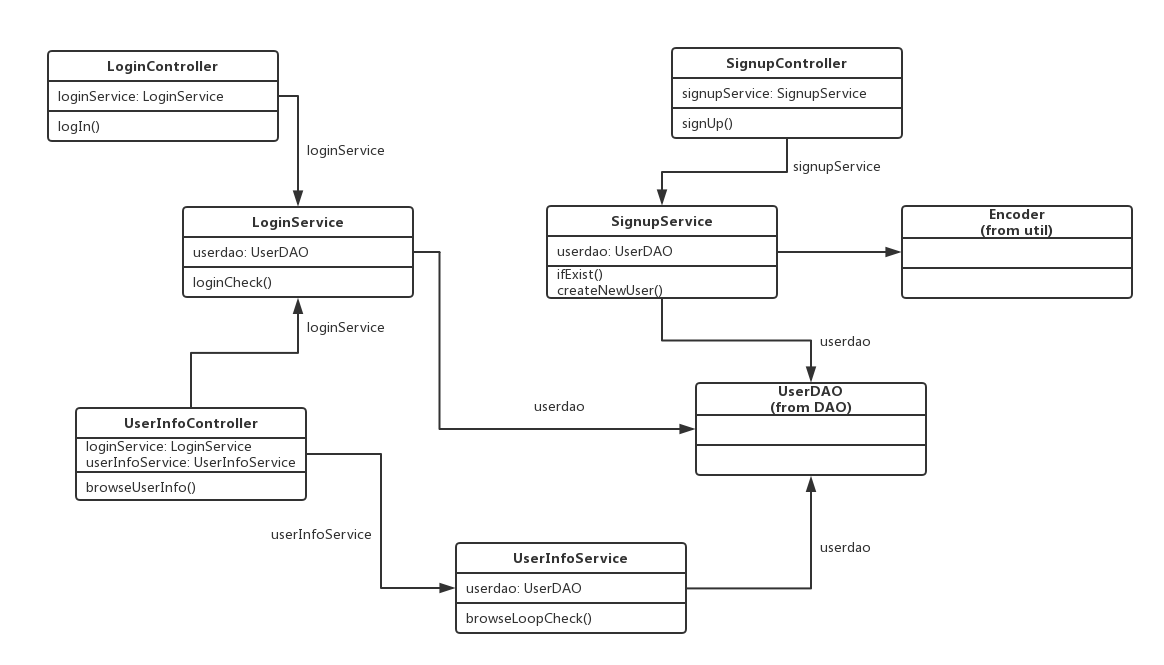


子系统架构如下：



## 类设计

* 1. **User相关**



3.1.1 LoginController

说明：接受登陆HTTP请求并在业务处理完毕后给予回复的控制类

属性：

loginService: LoginService 通过Spring Boot注解依赖注入的业务类

方法：

logIn(req: LoginRequest): IfSuccessResponse 接受HTTP请求，调用loginService的loginCheck()方法进行登陆业务处理，并在完成后给予回复，返回IfSuccessResponse响应类

3.1.2 LoginService

说明：进行登陆业务处理的服务类

属性：

userdao: UserDAO 通过Spring Boot 注解依赖注入的数据库交互类

方法：

loginCheck(username: String, password: String): User 进行用户登陆验证的业务方法，调用UserDao的 loginCheck()方法访问数据库用户信息来进行业务处理，返回登录成功的用户User类

3.1.3 SignupController

说明：接受注册HTTP请求并在业务处理完毕后给予回复的控制类

属性：

signupService: SignupService 通过 Spring Boot 注解依赖注入的业务类

方法：

signUp(req: SignupRequest): IfSuccessResponse 接受HTTP请求，调用signupService的ifExist()方法判断用户是否存在，根据结果调用signupService的createNewUser()方法创建新用户，返回IfSuccessResponse响应类

3.1.4 SignupService

说明：进行注册业务处理的服务类

属性：

userdao: UserDAO通过Spring Boot 注解依赖注入的数据库交互类

方法：

ifExist(name: String, identity: String): Boolean 判断目标用户名及身份信息在系统中是否存在，调用userDAO的checkExistList()方法访问数据库信息来进行业务处理，返回判断结果

createNewUser(username: String, password: String, identity: String, nickname: String): Boolean 创建指定信息的用户，调用Encoder类的静态方法encryptBasedDes()来对密码信息加密后，调用userDAO的addUser()方法向数据库写入信息，返回注册是否成功的结果boolean

3.1.5 UserInfoController

说明：接受查看用户信息HTTP请求并在业务处理完毕后给予回复的控制类

属性：

loginService: LoginService 通过Spring Boot注解依赖注入的业务类，用于在业务前进行权限检查

userInfoService: UserInfoService 通过Spring Boot注解依赖注入的业务类

方法：

browseUserInfo(req: BrowseRequest): IfSuccessResponse 接受HTTP请求，调用loginService的loginCheck()方法检查权限后，调用userInfoService的browseLoopCheck()方法获取环路信息，返回IfSuccessResponse响应类

3.1.6 UserInfoService

说明：进行查看用户信息的业务类（由于其他用户信息在登录时就已经返回，该类暂只用于查看环路信息）

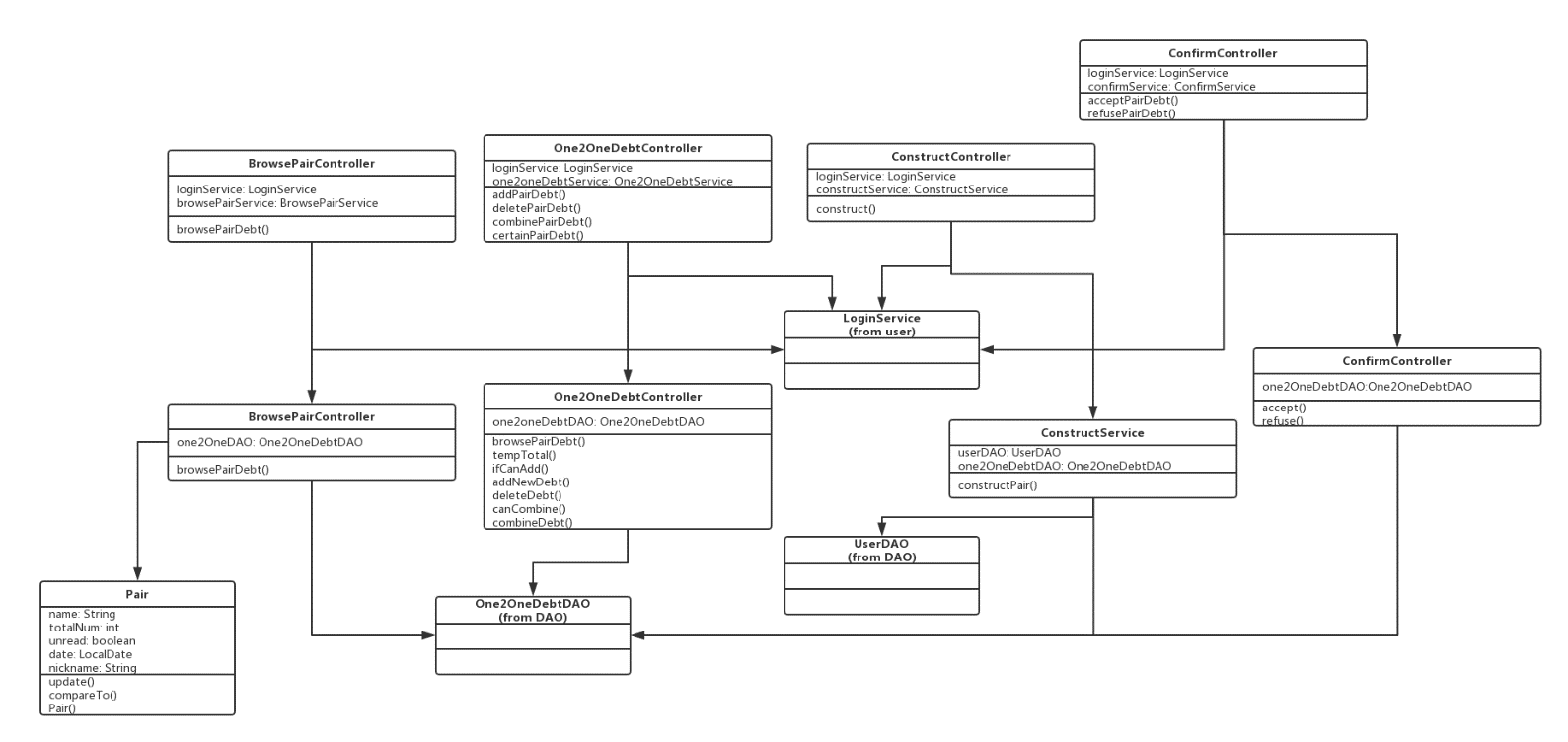
属性：

userDAO: UserDAO 通过Spring Boot 注解依赖注入的数据库交互类

方法：

browseLoopCheck(username: String): List<LoopCheck> 调用userDAO的 getLoopCheck()方法与数据库交互访问用户环路信息，返回LoopCheck环路类列表

* 1. **One2OneDebt相关**



* + 1. BrowsePairController

说明：接受查看用户结对列表HTTP请求并在业务处理完毕后给予回复的控制类

属性：

loginService: LoginService 通过Spring Boot注解依赖注入的业务类，用于在业务前进行权限检查

browsePairService: BrowsePairService 通过Spring Boot注解依赖注入的业务类

方法：

browsePairDebt(req:BrowseReques): IfSuccessResponse 接受HTTP请求，调用loginService的loginCheck()方法检查权限后，调用browsePairService的browsePair()方法获取结对信息，返回IfSuccessResponse响应类

* + 1. BrowsePairService

说明：进行查看用户所处结对关系列表的业务类

属性：

one2OneDAO: One2OneDebtDAO 通过Spring Boot 注解依赖注入的数据库交互类

方法：

browsePair(username: String): Map<String,Pair> 调用one2OneDAO的findAllDebts()方法访问数据库获取用户的所有一对一账目信息One2OneDebt后，处理封装成用于浏览结对关系的Pair类，返回结对关系对方用户名和Pair类的键值对Map<String,Pair>

* + 1. Pair

说明：适用于浏览用户结对关系的Model类，是对One2OneDebt实体类的组装

属性：

name: String 结对关系中对方用户名

totalNum: int 总金额，正代表资金从用户流出到对方

unread: Boolean 新操作表示，true代表存在未被确认的账目

date: LocalDate 关系下最新一笔账目的时间

nickname: String 对方的昵称

方法：

update(num: int, out: boolean, date: LocalDate, unread: boolean): void 用于封装Pair时更新信息的方法，根据输入的账目属性更新自身属性, 参数中out代表该笔资金是否是从用户流出到对方

compareTo(o: Pair): int：重写Object的比较，用于排序，使得存在新操作和时间较新的Pair能在使用Collections.sort（）方法排序时排在前端

Pair(name: String, totalNum: int, unread: boolean, date: LocalDate, nickname: String): 以属性为参数的构造函数

* + 1. ConstructController

说明：接受新增结对关系HTTP请求并在业务处理完毕后给予回复的控制类

属性：

loginService: LoginService通过Spring Boot注解依赖注入的业务类，用于在业务前进行权限检查

constructService: ConstructService 通过Spring Boot注解依赖注入的业务类

方法：

construct(req: ConstructRequest): IfSuccessResponse 接受HTTP请求，调用loginService的loginCheck()方法检查权限后，调用constructService的constructPair()方法进行结对请求添加，返回IfSuccessResponse响应类

* + 1. ConstructService

说明：进行发起结对请求功能的业务类

属性：

userDAO: UserDAO 通过Spring Boot 注解依赖注入的数据库交互类

one2OneDebtDAO: One2OneDebtDAO 通过Spring Boot 注解依赖注入的数据库交互类

方法：

constructPair(username: String, counterName: String, desc: String): One2OneDebt 首先调用userDAO的checkUser()方法寻找目标用户，若找到则继续，否则返回Null；通过one2OneDebtDAO的findDebts()方法检查用户与目标间的账目数量是否为0（即是否未建立关系），若是则继续，否则返回Null；最后调用one2OneDebtDAO的createDebt方法新建一笔资金为0的初始待确认账目，返回该笔账目One2OneDebt

* + 1. One2OneDebtController

说明：接受一对一记账账目相关操作HTTP请求并在业务处理完毕后给予回复的控制类

属性：

loginService: LoginService通过Spring Boot注解依赖注入的业务类，用于在业务前进行权限检查

one2OneDebtService: One2OneDebtService 通过Spring Boot注解依赖注入的业务类

方法：

addPairDebt(req: One2OneDebtRequest): IfSuccessResponse 接受HTTP请求，调用loginService的loginCheck()方法检查权限后，先调用one2OneDebtService的ifCanAdd()方法判断你是否能够添加，再调用one2OneDebtService的addNewDebt()方法在用户指定关系下新增一笔待添加账目，返回IfSuccessResponse响应类

deletePairDebt(req: One2OneSelectRequest): IfSuccessResponse 接受HTTP请求，调用loginService的loginCheck()方法检查权限后，调用one2OneDebtService的deletePairDebt()方法在用户指定关系下将指定负数账目设为待删除状态，返回IfSuccessResponse响应类

combinePairDebt(req: One2OneSelectRequest): IfSuccessResponse 接受HTTP请求，调用loginService的loginCheck()方法检查权限后，调用one2OneDebtService的canCombine()方法判断是否能进行合并，再调用one2OneDebtService的combineDebt()方法将目标复数账目标记为待合并并新增待确认合并结果，返回IfSuccessResponse响应类

certainPairDebt(req: CounterRequest): IfSuccessResponse 接受HTTP请求，调用loginService的loginCheck()方法检查权限后，调用one2OneDebtService的browsePairDebt()方法获取用户在目标关系下的左右账目，再调用tempTotal()方法简单计算总账数字以保证用户端信息同步，返回IfSuccessResponse响应类

* + 1. One2OneDebtService

说明：进行一对一记账账目相关功能的业务类，包括：更新账目（添加，删除，合并），浏览关系下账目

属性：

one2OneDebtDAO: One2OneDebtDAO 通过Spring Boot 注解依赖注入的数据库交互类

方法：

browsePairDebt(username: String, counter: String): List<One2OneDebt> 通过调用one2OneDebtDAO的findDebts()方法获取用户在目标关系下的所有账目，返回账目One2OneDebt的列表

tempTotal(list: List<One2OneDebt>, username: String): int 计算输入列表账目对于输入用户来说的资金数字总和，返回结果，该方法用于在用户请求关系下所有账目列表时简单更新总账数字

ifCanAdd(username: String, counter: String): Boolean 调用one2OneDebtDAO的findAcceptedDebts()方法，并根据据结果判断用户是否能在指定关系下添加账目，返回结果boolean

addNewDebt(username: String, counter: String, num: int, desc: String, ifUserOut: boolean): One2OneDebt 通过one2OneDebt的createDebt方法为用户向目标关系下新增一笔待添加账目，参数中的ifUserOut代表用户是否为账目起始节点，返回该账目One2OneDebt

deleteDebt(username: String, counter: String, list: String[]): List<One2OneDebt> 调用one2OneDebtDAO的checkStatus()方法检查目标账目列表中的项是否可删除，再分别调用one2OneDebtDAO的setStateStart()方法和setStateEnd()方法将账目标记为待删除（分别对应用户作为起始节点和终止节点），返回这些账目One2OneDebt列表

canCombine(username: String, counter: String): Boolean 调用one2OneDebtDAO的checkCombine()方法检查目标关系下的待确认账目数，判断是否能够发起合并操作，返回结果Boolean

combineDebt(username: String, counter: String, list: String[], desc: String): One2OneDebt 调用one2OneDebtDAO的setStatusStart()与setStatusEnd()方法将待合并项置为待合并状态，通过调用one2OneDebtDAO的createDebt()方法添加待合并结果账目，返回该结果账目One2OneDebt

* + 1. ConfirmController

说明：接受确认或拒绝账目HTTP请求并在业务处理完毕后返回回复的控制类

属性：

loginService: LoginService 通过Spring Boot注解依赖注入的业务类，用于在业务前进行权限检查

confirmService: ConfirmService 通过Spring Boot注解依赖注入的业务类

方法：

acceptPairDebt(req: ConfirmRequest): IfSuccessResponse 接受HTTP请求，调用loginService的loginCheck()方法检查权限后，调用confirmService的accept()方法进行确认业务，返回IfSuccessResponse响应类

refusePairDebt(req: ConfirmRequest): IfSuccessResponse 接受HTTP请求，调用loginService的loginCheck()方法检查权限后，调用confirmService的refuse()方法进行拒绝业务，返回IfSuccessResponse响应类

* + 1. ConfirmService

说明：进行确认拒绝相关功能的业务类

属性：

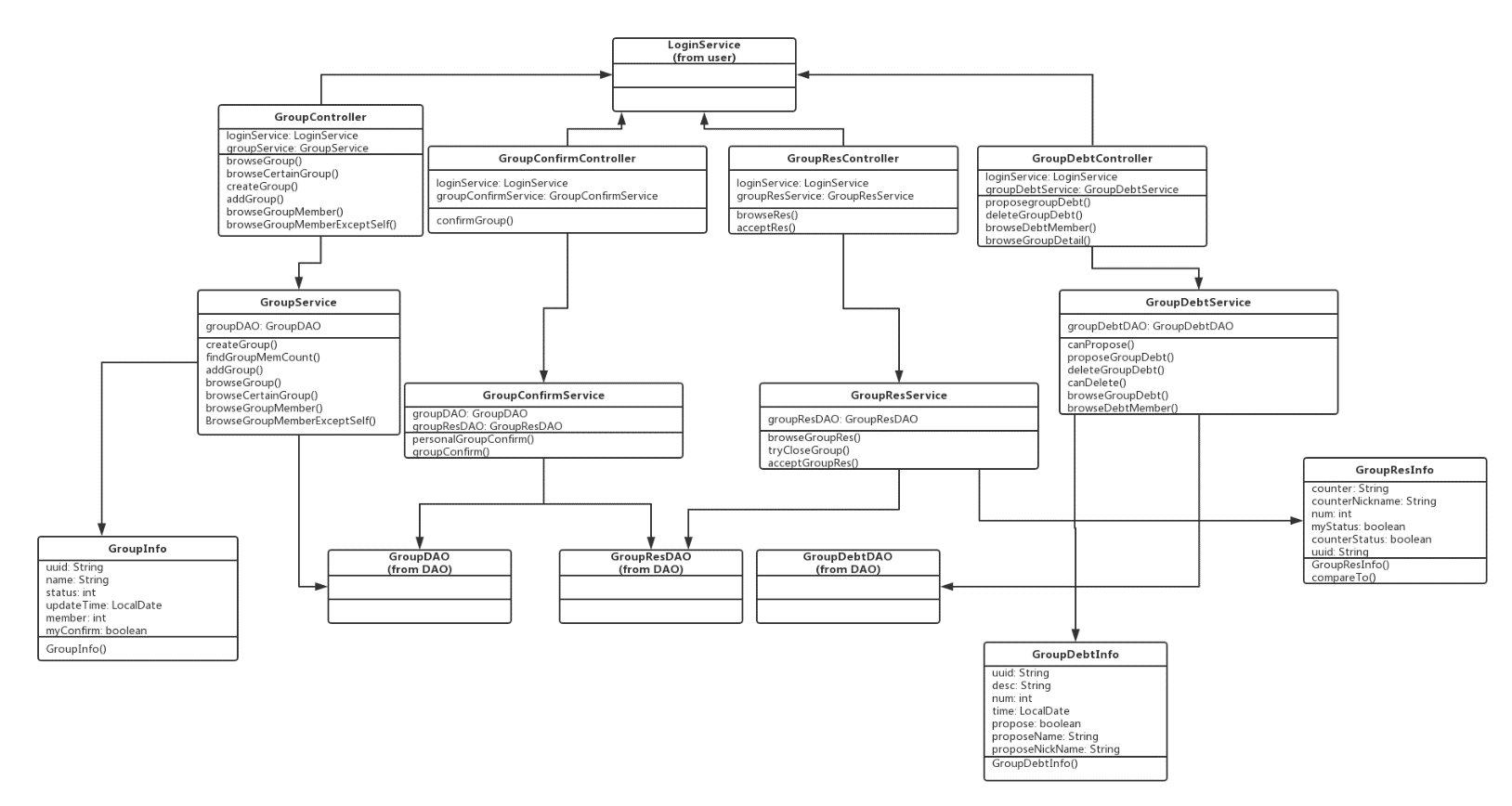
debtDao: One2OneDebtDAO 通过Spring Boot 注解依赖注入的数据库交互类

方法：

accept(username: String, countername: String, id: String, status: int): One2OneDebt 通过调用one2OneDebtDAO的getDebt()方法获取目标账目，判断确认是否合法，再根据其待确认状态调用one2OneDebt的setStatus(), deleteDebt()和confirmCombineDelete()方法进行账目确认更新，返回目标账目One2OneDebt

refuse(username: String, countername: String, id: String, status: int): One2OneDebt 通过调用one2OneDebtDAO的getDebt()方法获取目标账目，判断确认是否合法，再根据其待确认状态调用one2OneDebt的setStatus(), deleteDebt()和confirmCombineCancel()方法进行账目拒绝更新，返回目标账目One2OneDebt

* 1. **Group相关**



* + 1. GroupController

说明：接受群组相关HTTP请求，包括创建群组，加入群组，浏览群组，浏览群组成员，并在业务处理完毕后给予回复的控制类

属性：

loginService: LoginService 通过Spring Boot注解依赖注入的业务类，用于在业务前进行权限检查

groupService: GroupService 通过Spring Boot注解依赖注入的业务类

方法：

browseGroup(req: BrowseRequest): IfSuccessResponse 调用groupService的同名称方法进行查找用户加入的群组列表的业务

browseCertainGroup(req: GroupRequest): IfSuccessResponse 调用groupService的同名称方法进行查找用户加入的一个指定群组信息的业务

createGroup(req: CreateGroupRequest): IfSuccessResponse 调用groupService的同名称方法进行创建一个新群组的业务

addGroup(req: GroupRequest): IfSuccessResponse 调用groupService的同名称方法进行查找并加入一个群组的业务

browseGroupMember(req: GroupRequest): IfSuccessResponse 调用groupService的同名称方法进行查找已加入的指定群组的成员列表的业务

browseGroupMemberExceptSelf(req: GroupRequest): IfSuccessResponse 调用groupService的同名称方法进行查找已加入的指定群组的不包含自己的成员列表的业务（用于用户端选择新群组账目目标）

以上方法进行业务类调用前，均接受对应HTTP请求，并需要调用loginService的loginCheck()方法检查权限，方法最后均返回IfSuccessResponse响应类

3.3.2 GroupService

说明：进行群组相关功能的业务类，包括创建群组，加入群组，浏览群组，浏览群组成员

属性：

groupDAO: GroupDAO 通过Spring Boot 注解依赖注入的数据库交互类

方法：

createGroup(username: String, groupName: String): GroupInfo 调用groupDAO的createGroup()方法新建一个群组并调用自身的addGroup()方法将创建者加入群组，返回群组信息GroupInfo

findGroupMem(guid: String): int 调用groupDAO的findGroupMem()方法查找目标群组成员数量，返回数量

addGroup(username: String, guid: String): GroupInfo 首先调用自身findGroupMem()方法找到目标群组和群组人数，由人数和群组状态判断用户是否能加群，若是则调用groupDAO的addGroup()方法将用户加入群组，返回群组信息GroupInfo，否则返回Null

browseGroup(username: String): List<GroupInfo> 调用groupDAO的checkAllGroup()方法获取用户所加入的所有群组及用户确认状态，再通过调用groupDAO的findMultiGroupMem()获取这些群组的人数信息，将结果组装成群组信息GroupInfo的列表返回

browseCertainGroup(username: String, guid: String): GroupInfo 调用groupDAO的findGroupMem()方法获取目标群组成员数量，再调用groupDAO的getPersonalStatus()方法获取用户对于目标群组的确认状态，将结果组装成群组信息GroupInfo返回

browseGroupMember(uuid: String, username: String): List<User> 调用groupDAO的getGroupMember()方法获取指定群组的所有成员，再判断用户是否处于其中（以防不合法请求的跨权限查找，如获得自身并不处于其中的群组信息），返回成员信息User列表

browseGroupMemberExceptSelf(uuid: String, username: String): List<User> 同上，但是返回结果前从列表中排除请求用户自身，返回User列表

* + 1. GroupInfo

说明：适用于浏览群组信息以及用户对群组状态的Model类，是对Group类以及群组内人数、用户对群组状态数据的组装

属性：

uuid: String 对应群组的uuid

name: String 对应群组的名称

status: int 对应群组的状态

updateTime: LocalDate 对应群组的confirmTime

member:int 对应群组的人数

myConfirm: Boolean 用户对对应群组的确认状态，true为已确认

方法：

GroupInfo(group: Group, mem: int, myConfirm: boolean) 通过群组Group实体类及附加信息构造GroupInfo的构造方法

* + 1. GroupDebtController

说明：接受群组内账目相关HTTP请求，包括发起账目，删除账目，浏览账目对象成员，浏览群组账目详情，并在业务处理完毕后给予回复的控制类

属性：

loginService: LoginService 通过Spring Boot注解依赖注入的业务类，用于在业务前进行权限检查

groupDebtService: GroupDebtService 通过Spring Boot注解依赖注入的业务类

方法：

proposeGroupDebt(req: ProposeGroupDebtRequest): IfSuccessResponse 调用groupDebtService的同名称方法，进行发起群组内账目的业务

deleteGroupDebt(req: DeleteGroupDebtRequest): IfSuccessResponse 调用groupDebtService的同名称方法，进行删除群组内账目的业务

browseDebtMember(req: GroupRequest): IfSuccessResponse 调用groupDebtService的同名称方法，进行查看某一群组内账目之目标用户的业务

browseGroupDetail(req: GroupRequest): IfSuccessResponse 调用groupDebtService的同名称方法，进行查看群组内所有账目的业务

以上方法进行业务类调用前，均接受对应HTTP请求，并需要调用loginService的loginCheck()方法检查权限，方法最后均返回IfSuccessResponse响应类

* + 1. GroupDebtService

说明：处理群组内账目相关功能，包括发起账目，删除账目，浏览账目对象成员，浏览群组账目详情的业务类

成员：

groupDebtDAO: GroupDebtDAO 通过Spring Boot 注解依赖注入的数据库交互类

方法：

canPropose(username: String, uuid: String, targetList: String[]): Boolean 通过调用groupDebtDAO的checkPersonalStatus()方法，getDebtNum()方法，checkTargetInGroup()方法获取用户对群组确认状态，群组内账目数量，目标用户是否在群组中的信息，据此判断用户是否能在群组中新增账目，返回结果boolean

proposeGroupDebt(username: String, guid: String, desc: String, num: int, targetList: String[]): GroupDebt 先调用自身canPropose()方法判断是否能新增账目，若不能则返回Null，调用groupDebtDAO的createGroupDebt()方法创建群组，再调用groupDebtDAO的linkGroupDebt()方法将新账目与目标对象用户连起来，返回该笔账目GroupDebt

deleteGroupDebt(username: String, guid: String, deleteList: String[]): Boolean 调用自身canDelete()方法判断是否能删除账目，再通过调用groupDebtDAO的deleteGroupDebt()方法将目标账目删除，返回是否删除成功boolean

canDelete(username: String, uuid: String, targetList: String[]): Boolean 通过调用groupDebtDAO的checkPersonalStatus()方法和checkDeleteTarget()方法获取用户对群组状态和删除目标是否在群组中的信息，据此判断你是否能执行删除，返回结果boolean

browseGroupDebt(username: String, uuid: String): List<GroupDebtInfo> 调用groupDebtDAO的findGroupDebt()方法获取群组内所有群组内账目及其发起人信息，将其组装成GroupDebtInfo类列表返回

browseDebtMember(uuid: String): List<User> 调用groupDebtDAO的findGroupDebtMember()方法获取指定群组内账目的所有目标用户信息，返回用户User类列表

* + 1. GroupDebtInfo

说明：适用于用户浏览群组账目的Model类，是对GroupDebt类以及相关用户User类部分数据的组装

属性：

uuid: String 对应群组内账目GroupDebt的uuid

desc: String 对应群组内账目的描述

num: int 对应群组内账目的金额

time: LocalDate 对应群组内账目的时间

propose: Boolean 用户是否是该条账目的发起人

proposeName: String 发起人用户名

proposeNickName: String 发起人昵称

方法：

GroupDebtInfo(debt: GroupDebt, proposer: User, username: String) 使用GroupDebt及发起人User类作为参数的构造函数

* + 1. GroupConfirmController

说明：接受用户确认群组HTTP请求并在业务处理完毕后给予回复的控制类

属性：

loginService: LoginService 通过Spring Boot注解依赖注入的业务类，用于在业务前进行权限检查

groupConfirmService: GroupConfirmService 通过Spring Boot注解依赖注入的业务类

方法：

confirmGroup(req: GroupRequest): IfSuccessResponse 接受HTTP请求，调用loginService的loginCheck()方法检查权限后，调用groupConfirmService的personalGroupConfirm()方法进行对群组确认的业务，返回IfSuccessResponse响应类

* + 1. GroupConfirmService

说明：处理群组确认相关功能，包括用户确认群组，群组转为确认状态的业务类

成员：

groupDAO: GroupDAO 通过Spring Boot 注解依赖注入的数据库交互类

groupResDAO: GroupResDAO 通过Spring Boot 注解依赖注入的数据库交互类

方法：

personalGroupConfirm(username: String, uuid: String): Group 调用groupDAO的setPersonalGroupStatus()方法将用户对指定群组的状态置为确认，调用自身groupConfirm()尝试将群组变为确认状态，返回群组Group类

groupConfirm(group: Group, time: LocalDate): Boolean 调用groupDAO的checkIfAllConfirmed()方法确认群组所有成员的所有对群组确认情况，若并非全确认则返回false；调用groupDAO的confirmGroup()方法将群组置为确认状态；计算群组内所有账目信息得到每两个有关系的用户间的结算结果；调用groupResDAO的createRes()方法创建群组这些结算结果，调用groupResDAO的linkResToGroup()方法将结算结果连接到群组上，返回是否成功确认boolean

* + 1. GroupResController

说明：接受HTTP请求并在业务处理完毕后给予回复的控制类

属性：

loginService: LoginService 通过Spring Boot注解依赖注入的业务类，用于在业务前进行权限检查

groupResService: GroupResService 通过Spring Boot注解依赖注入的业务类

方法：

browseRes(req: GroupRequest): IfSuccessResponse 接受HTTP请求，调用loginService的loginCheck()方法检查权限后，调用groupResService的browseGroupRes()方法进行查看群组内与用户相关的结算结果的业务，返回IfSuccessResponse响应类

acceptRes(req: AcceptGroupResRequest): IfSuccessResponse 接受HTTP请求，调用loginService的loginCheck()方法检查权限后，调用groupResService的acceptGroupRes()方法进行群组结算结果确认结清的业务，返回IfSuccessResponse响应类

* + 1. GroupResService

说明：处理用户确认结清结算结果，所有结算结果结清后群组关闭的业务类

属性：

groupResDAO: GroupResDAO 通过Spring Boot 注解依赖注入的数据库交互类

方法：

browseGroupRes(username: String, uuid: String): List<GroupResInfo> 调用groupResDAO的getResListIn()方法和getResListOut()方法获取用户在群组内处于起始节点和结束节点的结算结果，并将结果组装成GroupResInfo，返回GroupResInfo类列表

acceptGroupRes(username: String, uuid: String, ifStart: boolean): GroupRes 根据用户处于结算结果的起始节点还是结束节点，选择调用groupResDAO的acceptGroupResIn()或acceptGroupResOut()方法，将用户对该结算结果的确认结清状态置为确认结清，并调用类自身tryCloseGroup()方法尝试关闭所属群组，返回结算结果GroupRes

tryCloseGroup(uuid: String): void 调用groupResDAO的getAllResStatusByResID()方法获取群组内所有结算结果的结清状态，根据此判断群组是否关闭，若全部结清则调用groupResDAO的closeGroup()方法关闭群组，无返回

* + 1. GroupResInfo

说明：此类为为用户浏览结算结果而设计的Model类，是对GroupRes以及相关用户User信息的组装

属性：

counter: String 对方用户名

counterNickname: String 对方昵称

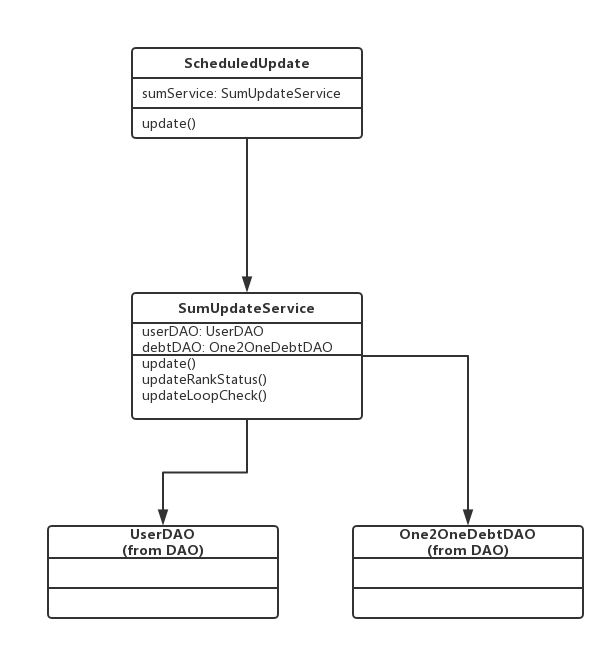
num: int 金额，如用户自身为资金流出方则为正，否则为负

myStatus: Boolean 用户自身对该结算结果是否确认结清

counterStatus: Boolean 对方对该结算结果是否确认结清

uuid: String 对应结算结果的uuid

* 1. **Sum相关**



3.4.1 ScheduledUpdate

说明：通过Spring Boot的 Scheduled注解控制执行定时任务的定时任务类

属性：

sumService: SumUpdateService 通过Spring Boot注解依赖注入的业务类，负责统计信息更新

方法：

update(): void 控制每日0:00进行sumService内业务的定时执行方法，调用sumService的update()方法开始业务执行

3.4.2 SumUpdateService

说明：执行更新用户总账目数字，用户所处环路检测，用户账目倾向功能的业务类

成员：

userDAO: UserDAO 通过Spring Boot 注解依赖注入的数据库交互类

debtDAO: One2OneDebtDAO 通过Spring Boot 注解依赖注入的数据库交互类

方法：

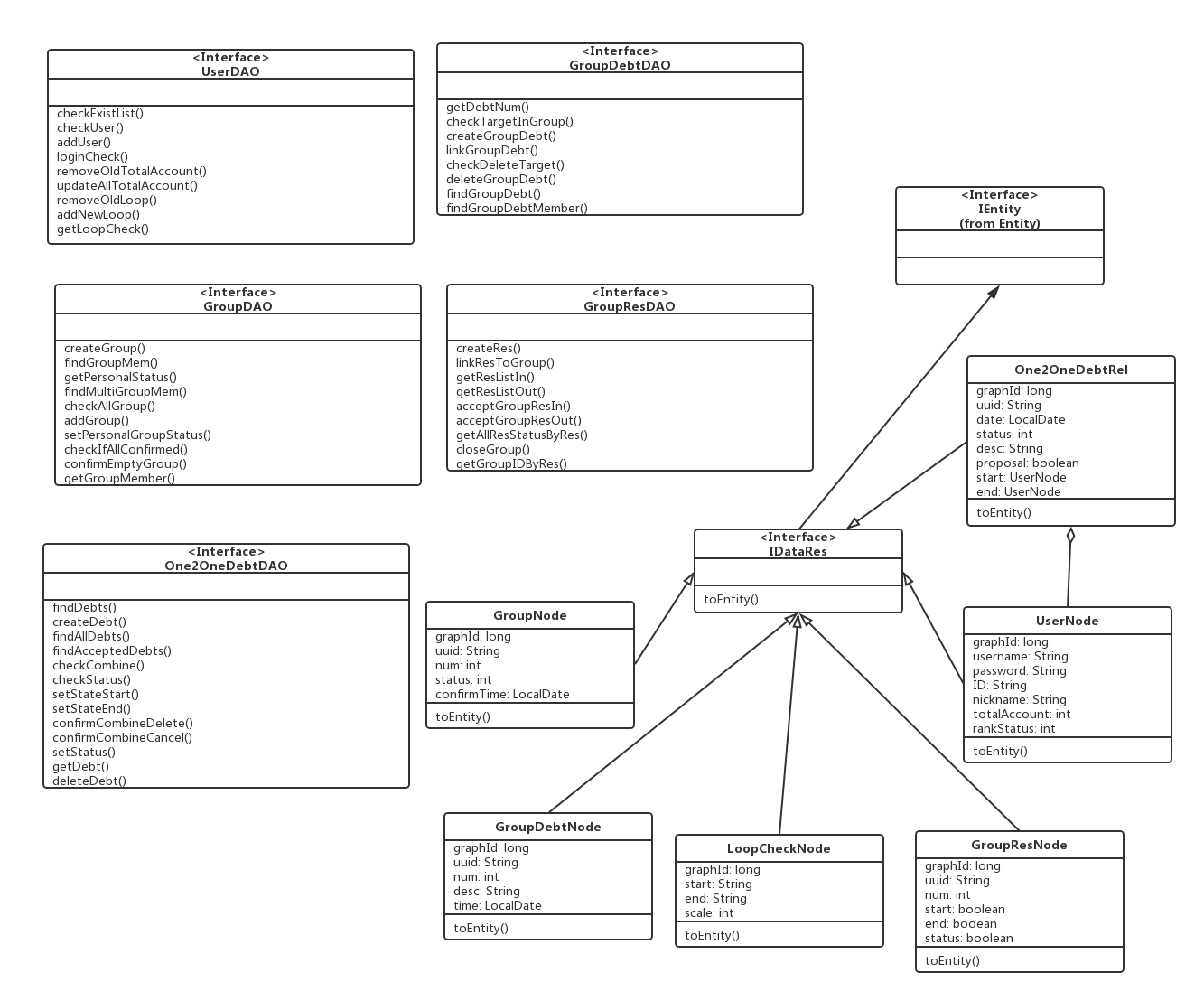
update(): void 调用userDAO的removeOldTotalAccount()方法清除所有用户的总账数字信息；调用userDAO的removeOldLoop()方法清除所有用户的环路检测旧信息；调用userDAO的updateAllTotalAccount()方法更新所有用户的总账数字信息；遍历所有用户，对每个用户调用类内updateRankStatus()方法和updateLoopCheck()方法

updateRankStatus(username: String): void 调用debtDAO的findALlDebts()方法获取用户的所有账目信息，根据账目的金额和时间信息计算出新的用户账目倾向数字，调用userDAO的setRank()方法写入更新信息

***注：计算方法有待修改***

updateLoopCheck(username: String): void 调用userDAO的addNewLoop()方法更新用户的环路检测信息

* 1. **DAO相关**



本模块中DAO类均为数据库接口，由注解的方式指定Query和返回结果，具体实现如连接池管理、OGM由Spring Data Neo4j实现接口。接口方法功能见上文具体使用时的调用说明。

3.5.1 UserDAO

说明：负责用户相关信息的数据持久接口

方法：

checkExistList(name: String, identity: String): int

checkUser(name: String): int

addUser(name: String, password: String, identity: String, total: int, rank: int, nickname: String): void

loginCheck(name: String, password: String): List<UserNode>

removeOldTotalAccount(): List<UserNode>

updateAllTotalAccount(): void

removeOldLoop(): void

setRank(username: String, rank: long): void

getLoopCheck(username: String): List<LoopCheckNode>

addNewLoop(username: String): void

3.5.2 GroupDAO

说明：负责群组相关信息的数据持久接口

方法：

createGroup(uuid: String, name: String, status: int, date: String): GroupNode

findGroupMem(uuid: String): GroupNodeInteger 注：GroupNodeInteger类为方便Query结果进行OGM映射而创建的中间结果类，下文中GroupNodeBoolean亦属于此

getPersonalStatus(username: String, uuid: String): Boolean

findMultiGroupMem(uuid: String[]): List<GroupNodeInteger>

checkAllGroup(username: String): List<GroupNodeBoolean>

addGroup(username: String, uuid: String, confirmed: boolean): int

setPersonalGroupStatus(username: String, currentStatus: boolean, guid: String, targetStatus: boolean): GroupNode

checkIfAllConfirmed(uuid: String): Boolean

confirmGroup(uuid: String, date: String): List<ConfirmFindDebt>

confirmEmptyGroup(uuid: String, date: String): int

getGroupMember(uuid: String): List<UserNode>

* + 1. One2OneDebtDAO

说明：负责一对一记账账目相关的数据持久接口

方法：

findDebts(username: String, counterUsername: String): List<One2OneDebtRel>

findAllDebts(username: String): List<One2OneDebtRel>

findAcceptedDebts(username: String, countername: String): int

checkCombine(username: String, countername: String): int

checkStatus(username: String, countername: String, id: String[], status: int): int

setStateStart(username: String, id: String[], status: int, proposal: boolean): List<One2OneDebtRel>

setStateEnd(username: String, id: String[], status: int, proposal: boolean): List<One2OneDebtRel>

confirmCombineDelete(username: String, countername: String): void

confirmCombineCancel(username: String, countername: String): int

setStatus(id: String, status: int): One2OneDebtRel

getDebt(id: String): One2OneDebtRel

deleteDebt(id: String): void

* + 1. GroupDebtDAO

说明：负责群组记账中群组内账目相关的数据持久接口

方法：

getDebtNum(uuid: String): int

checkPersonalStatus(username: String, uuid: String): int

checkTargetInGroup(usernames: String[], uuid: String): int

createGroupDebt(username: String, guid: String, uuid: String, desc: String, num: int, time: String): GroupDebtNode

linkGroupDebt(uuid: String, username: String[]): int

checkDeleteTarget(username: String, targetList: String[], uuid: String): int

deleteGroupDebt(targetList: String[]): void

findGroupDebt(uuid: String): List<GroupDebtNodeUserNode>

findGroupDebtMember(uuid: String): List<UserNode>

* + 1. GroupResDAO

说明：负责群组记账中群组确认后产生的结算结果相关的数据持久接口

方法：

createRes(start: String, end: String, uuid: String, num: int): GroupResNode

linkResToGroup(idList: String[], guid: String): int

getResListIn(username: String, uuid: String): List<GroupResNodeUserNode>

getResListOut(username: String, uuid: String): List<GroupResNodeUserNode>

acceptGroupResIn(username: String, uuid: String): GroupResNode

acceptGroupResOut(username: String, uuid: String): GroupResNode

getAllResStatusByResID(uuid: String): List<Boolean>

closeGroup(uuid: String): int

getGroupIDByRes(uuid: String): String

* + 1. IDataRes接口

说明：该接口表明为实体类对应的数据库内容映射结果

方法：

toEntity():IEntity 将映射类转换成实体类，返回一个对应的实体类

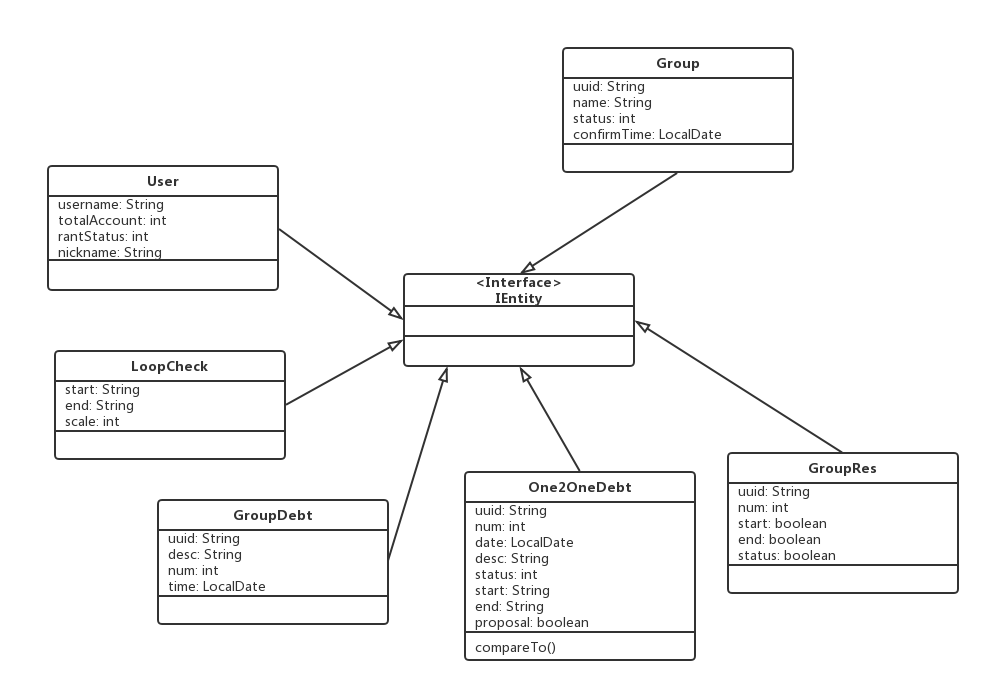
* + 1. 本模块中所有实现IDataRes接口的类

说明：这些类为实体类对应的映射类(Mapper)当实体数据从数据库中被取出，首先会被OGM映射成这些类，这些类带有数据库内ID，加密密码等不适宜参与业务运算的冗余信息，故需要转换成实体类后参加业务

映射类属性含义见对应实体类及[rBook概要设计规约](rBook概要设计规约.docx)数据库设计

方法：toEntity():IEntity 将映射类转换成实体类，返回一个对应的实体类

* 1. **Entity相关**



实体类主要是数据承载，无自身业务功能

3.6.1 IEntiy接口

说明：所有实体类实现的接口，无功能，方便映射类的直接转化

3.6.2 User

说明：用户实体（不包括密码，出于安全考虑，用户加密密码不带出数据库及登录模块以外的模块）

属性：

username: String 用户名

totalAccount: int 总账目数字，区分正负，正为资金流出

rankStatus: int 用户账目倾向

nickname: String 用户昵称

* + 1. One2OneDebt

说明：一对一账目实体

属性：

uuid: String 全局唯一标识

num: int 资金数目，正整型

date: LocalDate 账目时间

desc: String 账目描述

status: int 账目状态 0代表确认，1代表待添加，2代表待删除，3代表待合并，4代表待确认的合并后结果

start: String 起始节点用户名（资金流出方）

end: String 终止节点用户名

proposal: Boolean 当status不为0时有意义，代表使它成为待确认状态用户是否是start用户，用于判断确认权限

方法：

compareTo(o: One2OneDebt): int 重写的比较方法，用于排序时让账目能以status为主要排序键，date为次要排序键降序排序

* + 1. LoopCheck

说明：环路检测结果类

属性：

start: String 用户在环路中的前驱节点用户名

end: String 用户在环路中的后继ouji 节点用户名

scale: int 环路大小（路径长度）

* + 1. GroupRes

说明：群组确认后产生的结算结果类

属性：

uuid: String 全局唯一标识

num: int 金额，正整形（方向由相关的边表示）

start: Boolean 起始节点用户确认结算状态

end: Boolean 结束节点用户确认结算状态

status: Boolean 该结果是否被结清（双方均确认结清则为结清）

* + 1. GroupDebt

说明：群组内账目类

属性：

uuid: String 全局唯一标识

desc: String 账目描述

num: int 账目金额，正整形

time: LocalDate 账目发起时间

* + 1. Group

说明：群组类

属性：

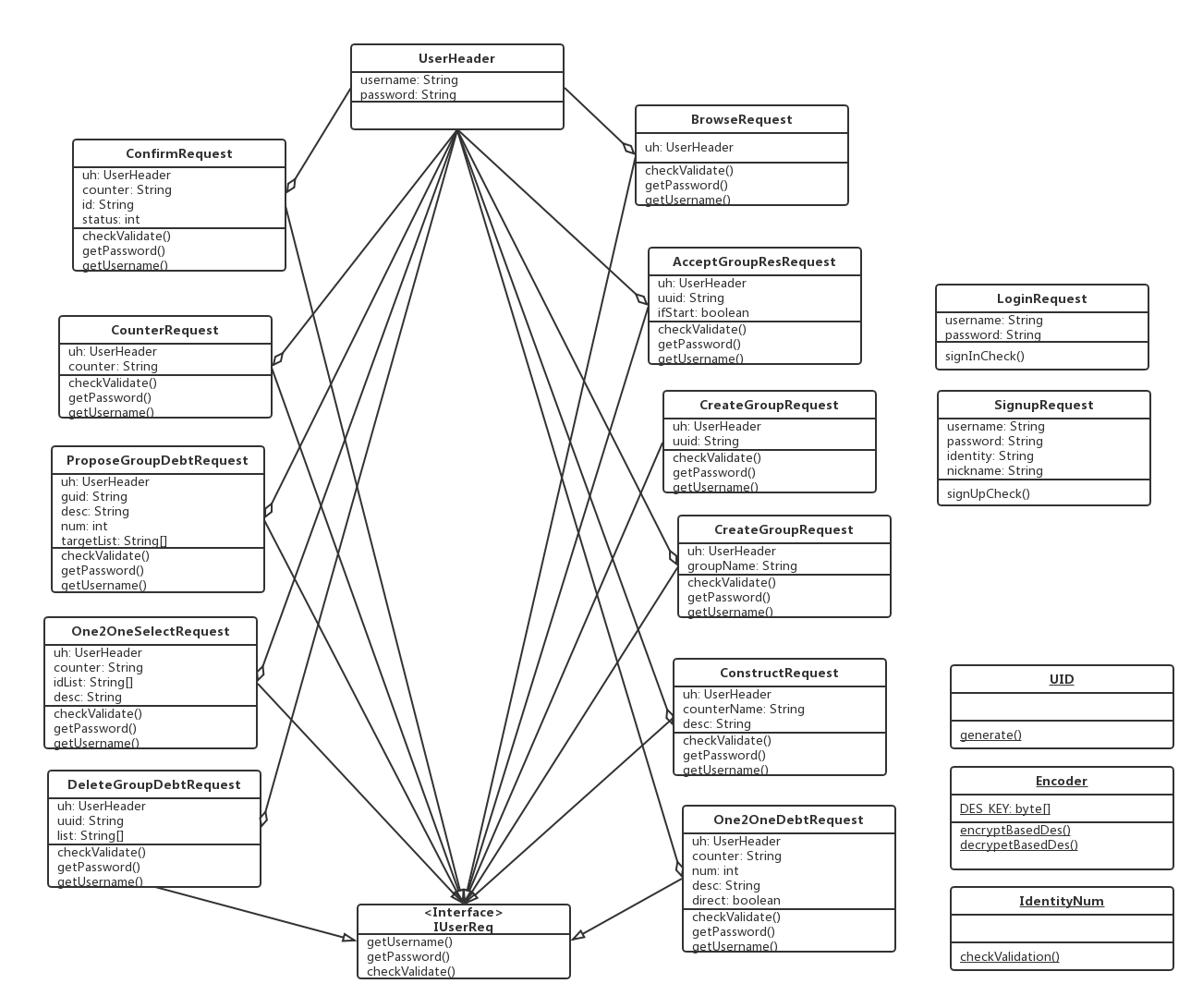
uuid: String 全局唯一标识

name: String 群组名

status: int 群组状态，0代表活跃，1代表已确认，2代表关闭

confirmTime: LocalDate 当群组状态为0，则代表群组创建时间；否则代表群组确认时间

* 1. **Common & Util相关**



本模块中的Request类为HTTP请求所用的类，具体对应功能和含义见[rBook概要设计规约](rBook概要设计规约.docx)接口设计

3.7.1 UserHeader

说明：用户请求中必须要包含的用户头，用于权限和身份验证

属性：

username: String 用户名

password: String 密码

3.7.2 IUserReq接口

说明：所有除登陆和注册外的HTTP请求类均为用户请求，需要实现该接口，并包括一个UserHeader作为成员

方法：

getUsername(): String 获得用户头的用户名

getPassword(): String 获得用户头的密码

checkValidate(): Boolean 检查请求格式合法性，返回结果boolean

* + 1. UID

说明：静态工具类，负责系统中所有全局唯一标识uuid的生成

方法：

generate(): String 静态方法，返回一个生成的uuid

* + 1. IdentityNum

说明：静态工具类，负责注册时验证用户身份证信息格式合法性

方法：

checkValidation(idNum: String): Boolean 返回合法性Boolean

* + 1. Encoder

说明：静态工具类，负责对用户密码进行简单的加密和解密

属性：

DES\_KEY: byte[] 常量，秘钥组

方法：

encryptBasedDes(data: String): String 加密方法，返回加密后的内容

decryptBasedDes(cryptData: String): String 解密方法，返回解密后的内容