## 进度同步

**注：该模块**主要对任务分发的数据进行汇总,自定义汇总定制,问题记录等功能(其中包含动态表的设计,已完成对其的增删改查,对任务的分发,已暂不使用)

**实现思路**

1. **项目汇总(如果看不懂sql可自己重写,sql可在调式时输出得到):**
2. 根据用户选择的条件,是否需要固定表头,汇总部门或者汇总某部分人的

任务来进行条件的构建组装sql

1. 查询今天的任务(所需字段请看需求文档)
2. 查询明天的任务(所需字段请看需求文档)
3. 构建封装对象并组装html
4. **自定义定制汇总(该汇总自定义的定制条件较多,注意构建条件及条件的处理)**
5. 编写一个线程任务(TaskRunner)每隔500毫秒执行一次,查询用户定制的汇总规则(executeTask)(线程的暂定开始等操作已写方法,但没有向consumer提供api),注意查询汇总规则做了缓存,不会每次都会查数据库,假如直接在数据库上修改规则,是不能生效的
6. 通过数据库表的上次执行时间与自定义定制的规则(如每分钟或者每天几点调用)判断该规则是否可以执行,如果可以执行,解析自定义的规则并更新执行时间(注意不能抛出异常中断线程,把错误输出控制台忽略错误且更新执行时间,不管执行成功或者失败)
7. 调用customizeCollect方法汇总,并解析执行自定义规则构建查询条件,查询固定的表头汇总,自定义表头汇总,手动的对自定义汇总再进行汇总,部门汇总
8. 构建好汇总对象,根据对象构建出html内容(注意构建html比较繁琐,代码较多,但并不算杂,相对有点难懂的是查询自定义定制的表头内容查询及自定义定制的汇总,sql相对比较复制,嵌套查询较多,但汇总效率会高很多)
9. **问题记录**
10. 主要是对问题的录入及问题的跟踪处理,流程为: 问题提出->问题受理人受理

所有只提供了问题的增删改查,及受理接口,相对简单,业务也没有特别的处理方式,就不多做解释了.

动态表的设计及思路就不多说了,因为没有用到,因表头不确定,用的是Cglib构建动态对象再转json.以下是表设计(注意因为为了提高动态表的查询效率,该模块与任务分发模块同用一个数据库,并没有分库,否则导致sql查询失败)

**task\_customize** 自定义定制化汇总

**task\_problem** 问题

**task\_problem\_type** 问题类型

以下为动态表设计:

**task\_project** 项目

**task\_project\_range** 项目可见范围

**task\_table** 项目表

**task\_field** 表列

**task\_row** 表行

**task\_grid** 行列值

**task\_val** 值

**Task** 任务

遗留功能:

内部协助模板,内部协助单记录