# 定时器

1. 采用ThreadPoolTaskScheduler

//线程池容量

executor.setPoolSize(20);

//等待时长

executor.setAwaitTerminationSeconds(60);

1. 配置表mysql

Commoditycenter.tasktimer

Keyname 任务名,不允许重复

delay间隔

startime

endtime

lastruntime

lastrunresult

url

//格式1{"data":null,"errcode":0,"msg":null}

//格式2{"data":{"user":12724},"errcode":0,"msg":null}

//格式3{"data":{"user":12724,"msg":[{"userid":123,"flag":"xxx","msg":"通知消息"},{"userid":124}]},"errcode":0,”msg”:null}

返回格式{data,errcode,msg},

如果data里面有key=user,那么会将消息发送至user对应值,

handleurl 如果有配置,会将url接口返回中的data.msg发送到处理接口

useridlist 通知的user.id

remark

Isinvalid 是否有效

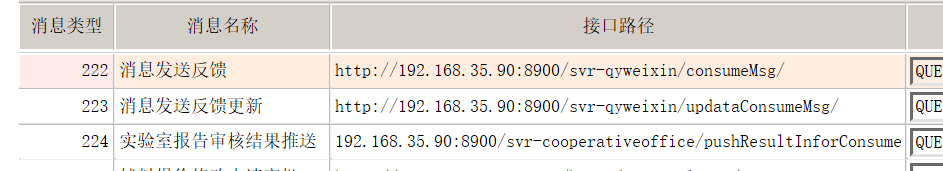
3.日志记录

Commoditycenter.tasktimerlog

3.日志记录

Commoditycenter.tasktimer4mq

MQ代码片断



    @RequestMapping("/pushResultInforSendMsg")

    public Result<?> pushResultInforSendMsg(Integer lbid, Integer sygzid) {

        JSONObject jo=new JSONObject();

        jo.put("msgtypeid", 224);

        JSONObject data=new JSONObject();

        data.put("lbid", lbid);

        data.put("sygzid", sygzid);

        jo.put("msginfo", data);

        return orderconsumer.createCallApiMsg(JSON.toJSONString(jo));

}

/\*\*

 \* 采用消息发送报告审核时候推送的信息---消费

 \* @param data

 \* @return

 \*/

    @RequestMapping("/pushResultInforConsume")

    public Result<?> consumeMsg(@RequestBody JSONObject data) {

        Integer lbid=data.getInteger("lbid");

        Integer sygzid=data.getInteger("sygzid");

        if (lbid == null) {

            return ResultUtil.error(5000, "参数不能为空");

        }

        return laboratoryInforpushService.pushResultInfor(lbid, sygzid);

    }

    public Result<?> sendMsg(JSONObject data) {

        JSONObject jo=new JSONObject();

        jo.put("msgtypeid", 222);

        jo.put("msginfo", data);

        return orderconsumer.createCallApiMsg(JSON.toJSONString(jo));

}

/\*\*

     \* 发起消息 的消费者

     \* @param data

     \* @return

     \*/

    @RequestMapping("/consumeMsg")

    public JSONObject consumeMsg(@RequestBody MsgdataDTO data) {

        return wxMessageService.consumeMsg(data);

    }

    public Result<?> upadteMsg(JSONObject data) {

        JSONObject jo=new JSONObject();

        jo.put("msgtypeid", 223);

        jo.put("msginfo", data);

        return orderconsumer.createCallApiMsg(JSON.toJSONString(jo));

    }

    /\*\*

     \* 发送更新消息 的消费者

     \* @param data

     \* @return

     \*/

    @RequestMapping("/updataConsumeMsg")

    public JSONObject updataMsg(@RequestBody MsgdataDTO data) {

        return wxMessageService.updataConsumeMsg(data);

    }