# 92\_高阶实战:基于分布式任务调度框架 xxl-job 优化 Spring Task 模块

**儒猿架构官网上线**,内有石杉老师架构课最新大纲,儒猿云平台详细介绍,敬请浏览

官网: www.ruyuan2020.com (建议 PC 端访问)

#### 1、开篇

上节课程先回顾了教师列表的缓存代码,发现只要满足缓存条件,并且缓存中有数据就会从缓存中获取对应的教师信息。因此,需要在 TaskService 中的 updateTeacherScore 方法上打上 EvictCache 的注释,说明要清除的缓存以及指定清除缓存中所有元素。这样当计划任务定时执行的时候,就会因为 EvictCache 注释的加持而自动清除缓存的内容。当查询教师列表时,发现缓存内容为空就会从数据库中获取教师的最新数据。本节课会使用分布式任务调度框架 XXL-Job 优化之前的 Spring Task,将教师的教学数和积分进行更新。今天课程的内容包括以下几个部分:

- xxl-job 的概要介绍
- xxl-job 在系统中的应用和配置

# 2、XXL-Job 概要介绍

XXL-JOB 是一个分布式任务调度平台,其核心设计目标是开发迅速、学习简单、轻量级、易扩展。现已开放源代码并接入多家公司线上产品线,开箱即用。在众多 XXL-Job 平台的特征中,有如下几条需要关注的:

- 1、使用简单: 支持通过 Web 页面对任务配置,降低操作任务的难度;
- 2、动态: 支持动态修改任务状态、启动/停止任务,以及终止运行中任务,即时生效;
- 3、调度中心 HA (中心式): 调度采用中心式设计,"调度中心"自研调度组件并支持集群部署,可保证调度中心 HA;

- 4、执行器 HA(分布式): 任务分布式执行,任务"执行器"支持集群部署,可保证任务执行 HA:
- 5、注册中心: 执行器会周期性自动注册任务,调度中心将会自动发现注册的任务并触发执行。同时,也支持手动录入执行器地址:
- 6、弹性扩容缩容:一旦有新执行器机器上线或者下线,下次调度时将会重新分配任务:
- 7、触发策略:提供丰富的任务触发策略,包括: Cron 触发、固定间隔触发、固定延时触发、API(事件)触发、人工触发、父子任务触发;

其他的特性这里不赘述了,有兴趣的同学可以去官网查看。

# 3、XXL-Job 在系统中的应用和配置

为了使用 XXL-Job 进行分布式任务调度的开发,需要引入相应的依赖。如图 1 所示,在项目的 pom.xml 文件中加入 dependency,设置 gorupld 为 com.xuxueli,



图 1

然后需要修改对应的配置文件,如图 2 所示,在 resources 目录的 xxl-job.properties 文件中会集中配置 xxl-job 信息。我们从上往下依次解释。

 xxl.job.admin.addresses 配置的是调度中心部署跟地址,作为选填项。如调度中心 集群部署存在多个地址则用逗号分隔。执行器将会使用该地址进行"执行器心跳注 册"和"任务结果回调",为空则关闭自动注册。目前儒猿团队给大家配置的是 http://127.0.0.1:8091/xxl-job-admin,使用的是 8091 端口。

- xxl.job.executor.appname 是执行器的名称,执行器心跳注册分组依据,为空则关闭自动注册,也就是当前项目在调度中心注册的名称。
- xxl.job.executor.ip 是执行器 IP [选填]: 默认为空表示自动获取 IP,多网卡时可手动设置指定 IP,该 IP 不会绑定 Host 仅作为通讯实用;地址信息用于 "执行器注册"和 "调度中心请求并触发任务";
- xxl.job.executor.port 执行器端口号 [选填]:小于等于 0 则自动获取;默认端口为 9999,单机部署多个执行器时,注意要配置不同执行器端口;
- xxl.job.executor.logpath 是 执行器运行日志文件存储磁盘路径 [选填]: 需要对该路径拥有读写权限;为空则使用默认路径;

```
▶ ■ handler
                                                    ### xxl-job admin address list, such as "<u>http://address" or "http://address8l.http://address82"</u>
### 调度中心器響跟地址 [选項]: 如调度中心集群部署存在多个地址则用班号分隔。执行器将会使用该地址进行"执行器心跳注册"和"任务结果回调"; 为空则关闭自动注册:
xxl.job.admin.addresses=http://2127.0.e.l.1893/xxl-job-admin
   ▶ ms jms
   ▶ 🛅 mapper
                                                    service
   ▶ utils
   ▶ □ validator
   ▼ 🖿 xxljob
resources
                                                    ### xxl-job, access token
### 执行器通讯TOKEN [选填]: 非空时启用;
xxl.job.accessToken=
 mapper mapper
   application.properties
    jdbc.properties
                                                    ## xxl-job log path ### xxl-job log path ### k/行器运行日志文件存储磁盘路径 [选填] : 需要对该路径拥有读写权限; 为空则使用默认路径; ### 执行器日志文件保存天数 [选填] : 过期日志自动清理,限制值大于等于3时生效; 否则, 如-1, 关闭自动清理功能; xxl-job.executor.logpath=/home/addim/little-project-spring/log/xxlJob ### xxl-job log retention days xxl.job.executor.logretentiondays=30
    a logback.properties
    a logback.xml
   amybatis-config.xml
    🏭 spring-jdbc.xml
    apring-jms.xml
    apring-task.xml
   apring-web.xml
     spring-xxl-job.xml
xxl-job.properties
```

#### 图 2

看完了配置再看看任务的实现是如何完成的。如图 3 所示,在 spring/xxljob 下面 创建 XxlJobHandler 的类,这个类可以作为一个 xxl-job 的任务被 xxl-job 的分布式 调度平台调度使用。在这个类中创建一个 updateTeacherScore 的方法,这个方法 是用来处理定时更新教师评分的任务。在该方法上面打上@XxlJob 的注释 updateTeacherScoreAndTeacherCount,同时也加入了 CacheEvict 的注释说明在 更新教师信息的时候会将 teacher 缓存全部清除。方法体中就是任务要执行的内容:

获取教师列表,获取已经完成的订单列表然后将订单更具教师 Id 存入 map 中,接着就是更新老师的评分信息。重新计算教师评分和重新计算教师授课总数。

```
annotion
        ► 🖿 cache
        constants
        ▶ ☐ controller
        ▶ ☐ dao
        ▶ 🛅 dto
        ▶ ☐ enums
        ▶ b event
                                                                                                       orderListByTeacherId.add(order);
orderMap.put(order.getTeacherId(), orderListByTeacherId);
        exception
        ▶ Imandler
                                                                                               }
List<Comment> updateCommentStatusList = new CopyOnWriteArrayList<>();
List<Order> updateOrderCountedList = new CopyOnWriteArrayList<>();
        ▶ interceptor
                                                                                                     st=Order> updateOrderCountedList = new CopyOnWriteArrayList<>();
更新教师序令信息
「Teacher teacher: teacherList) {
List=Order> orderListByTeacherId = orderMap.get(teacher.getId());
if (!CollectionUtils.isEmpty(orderListByTeacherId)) {
AtomicInteger orderCount = new AtomicInteger(initialValue: 0);
AtomicInteger commentCount = new AtomicInteger(initialValue: 0);
AtomicInteger commentCount = new AtomicInteger(initialValue: 0);
OrderListByTeacherId.forFach(order -> {
orderCount.addAndGet(delfa: 1);
Comment comment = order.getComment();
if (!StringUtils.isEmpty(order.getCommentId()) && CommentStatusEnum.NON_CHECKED.getCode().equals(comment.getScore());
comment.comment count.addAndGet(delfa: 1);
scoreTotal.addAndGet(comment.getScore());
comment estStatus(CommentStatusEnum.CHECKED.getCode());
updateCommentStatusList.add(comment);
}
        ▶ b jms
        ► 🛅 mapper
        ▶ service
        ► 🖿 task
        utils
        ▶ □ validator
         xxljob

    XxlJobHandler

▼ 📭 resources
       mapper
        application.properties
         捐 jdbc.properties
                                                                                                                       }
order.setCounted(YesOrNoEnum.YES.getCode());
updateOrderCountedList.add(order);
        a logback.properties
         Jogback.xml
                                                                                                               });
BigDecimal teacherCount = new BigDecimal(teacher.getTeachingCount());
// 重新计算教育评分
         mybatis-config.xml
         apring-jdbc.xml
```

图 3

### 4、总结

本节课主要介绍了 XXL-Job 的分布式任务调度架构以及特性。并且通过依赖项、基本配置信息以及应用三个方面对其进行了描述,看看是如何改造当前的 Spring task 的。

下节课,进入测试阶段,回归测试教师评分信息功能。这里将<u>代码</u>给大家,下期见,拜拜。

友情提示:本章讲述代码只是部分核心代码,完整代码请查阅文末链接中代码,谢谢。