**轻量级分布式任务调度框架**

**使用说明文档**

# 历史记录

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **版本** | **编制** | | **审核** | | **批准** | | **更改记录** |
| **发布人** | **日期** | **审核人** | **日期** | **批准人** | **日期** |
| V0.1 | 潘洪亮 | 2016.3.2 |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

# 目录

[历史记录 2](#_Toc444702176)

[目录 3](#_Toc444702177)

[1 分布式作业lts 4](#_Toc444702178)

[1.1 系统概述 4](#_Toc444702179)

[1.2 开发环境 4](#_Toc444702180)

[1.2.1 所需环境 4](#_Toc444702181)

[1.2.2 配置本地仓库 4](#_Toc444702182)

[1.2.3 编译 5](#_Toc444702183)

[1.2.4 运行 5](#_Toc444702184)

[1.3 部署 6](#_Toc444702185)

[1.4 运行 6](#_Toc444702186)

[1.5 开发业务作业 6](#_Toc444702187)

[1.5.1 自动扫描,派发作业 6](#_Toc444702188)

[1.5.2 Lts-admin界面手动添加作业 8](#_Toc444702189)

[1.5.3 Sql语句插入作业 8](#_Toc444702190)

[2 配置中心 8](#_Toc444702191)

[3 Mq rpc 8](#_Toc444702192)

[4 消息推送 8](#_Toc444702193)

[5 缓存 8](#_Toc444702194)

# 分布式作业lts

## 系统概述

<https://github.com/ltsopensource/light-task-scheduler>

readme.md 看几遍理解概念原理

light-task-scheduler\docs\OLD\_README.md 这个里边也有些重要信息

我的fork版本

https://github.com/hongliangpan/light-task-scheduler

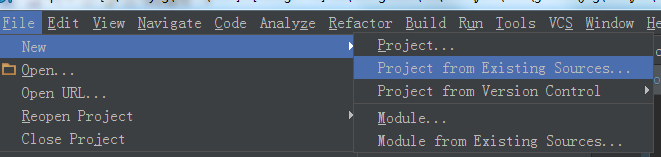
## 开发环境

### 所需环境

Zookeeper或redis作为注册中心【启动zkServer.cmd】

Mysql 数据库

Git、Idea，导入工程，选择父pom文件



### 配置本地仓库

父pom.xml

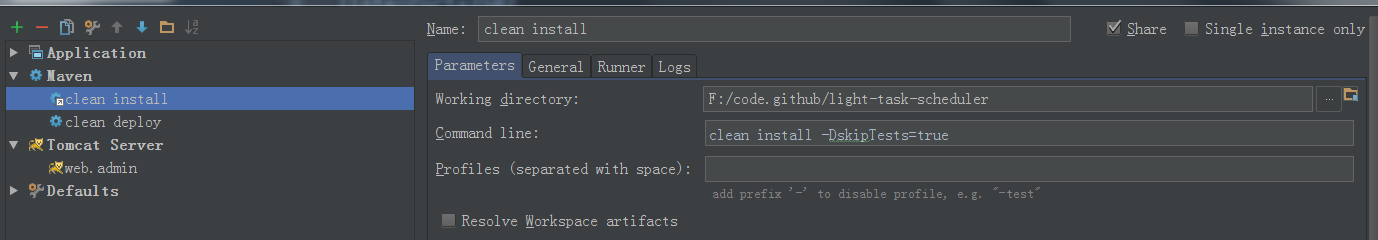
|  |
| --- |
| <distributionManagement>  <repository>  <id>glodon-nexus-releases</id>  <name>Glodon Nexus Release Repository</name>  <url>http://maven.glodon.com/nexus/content/repositories/releases/</url>  </repository>  <snapshotRepository>  <id>glodon-nexus-snapshots</id>  <name>Glodon Nexus Snapshot Repository</name>  <url>http://maven.glodon.com/nexus/content/repositories/snapshots/</url>  </snapshotRepository>  </distributionManagement> |

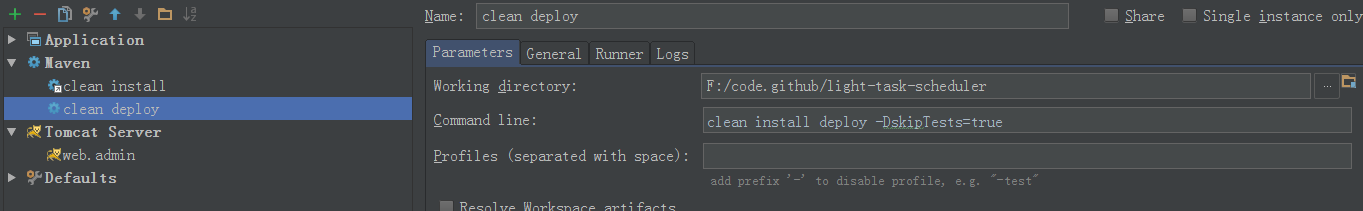
### 编译

clean install

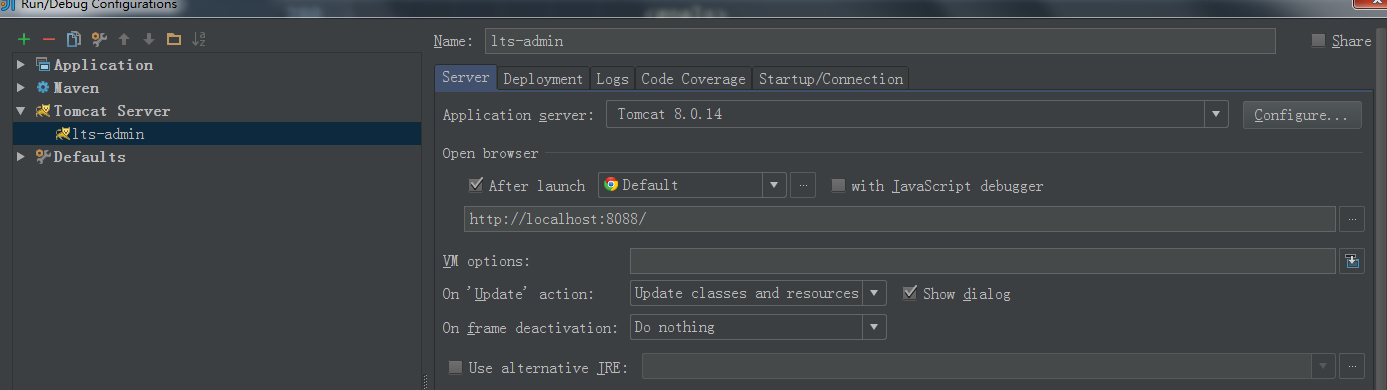
发布到本地仓库

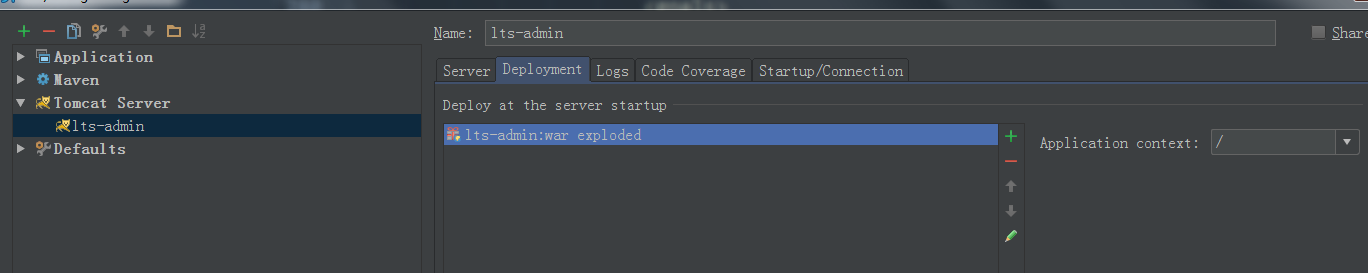
clean install deploy -DskipTests=true





Web.admin





### 运行

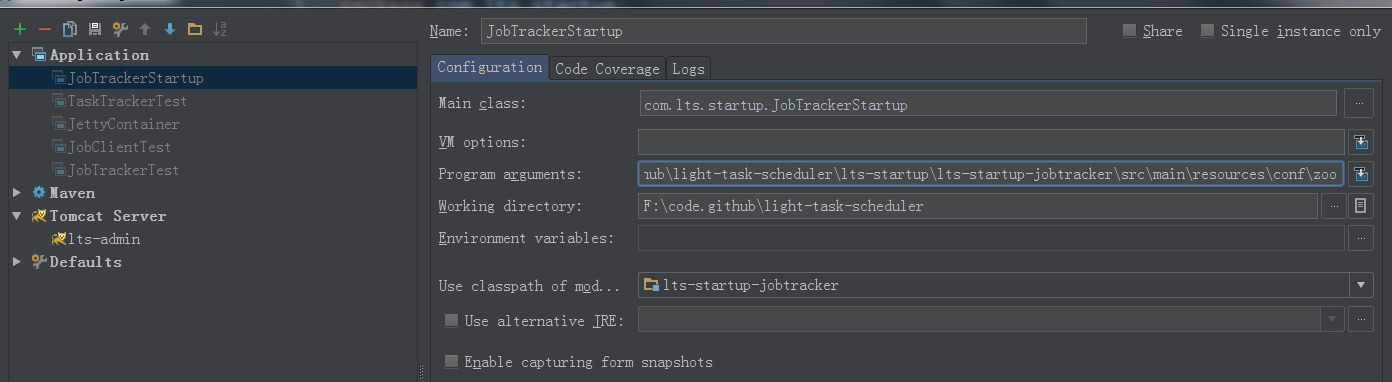
1. 修改配置文件，如数据库连接参数

conf/zoo/jobtracker.cfg

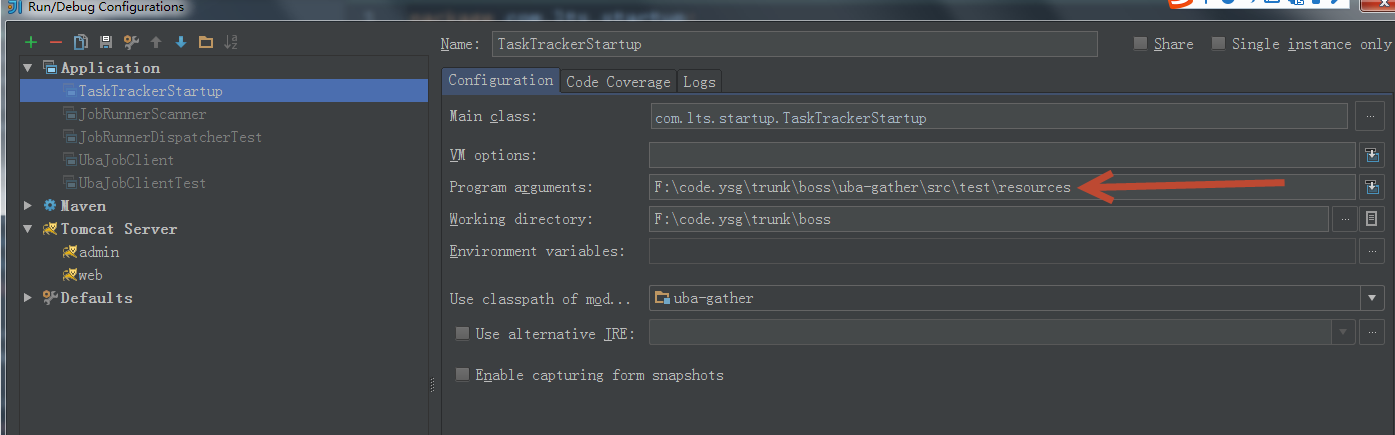
lts-admin.cfg

1. 运行 tomcat lts-admin
2. 运行JobTrackerStartup

第一个参数：为配置文件目录



1. TaskTrackerStartup+ tasktracker.cfg拷贝到业务工程运行



## 部署

运行 build\_all.cmd

lts推荐的部署方式 jobtracker独立部署 jobclient集成到业务模块 tasktracker也集成

## 运行

|  |
| --- |
| sh ./bin/jobtracker.sh zoo start  sh ./bin/lts-admin.sh start  sh ./tasktracker/bin/tasktracker.sh start |

## 开发业务作业

### 自动扫描,派发作业

|  |
| --- |
| // 作业开发部署  /\*  自动扫描作业，并派发[根据注解的type参数进行派发]  1. 配置JobRunnerDispatcher为JobRunner  tasktracker.cfg文件中  # JobRunner 任务执行类  jobRunnerClass=com.lts.tasktracker.jobdispatcher.JobRunnerDispatcher  2. 开发新的作业  //添加 类注解 @RunnerTask(type= "type")  @JobRunnerAnnotation(type= "type1")  public class GatherMetricJobRunner implements JobRunner  3.自动扫描作业JobRunnerDispatcher,配置扫描的package  static {  try {  JobRunnerScanner.scan("com.glodon.ysg", JOB\_RUNNER\_MAP);  } catch (Exception e) {  LOGGER.error(e.getMessage(),e);  }  }  4. 自行开发的作业，把jar包，放入tasktracker\lib  重启tasktracker，就自动运行  5. 配置参数  # JobRunner 任务执行类  jobRunnerClass=com.lts.tasktracker.jobdispatcher.JobRunnerDispatcher  # 自动扫描作业包，逗号分隔  jobRunnerScannerPackages=com.glodon.ysg,com.lts.job  # 是否内嵌JobClient  isEmbedJobClient=true  # 提交作业类，初始化一些作业，可以通过admin-web修改作业信息  jobSubmitterClass=com.lts.tasktracker.jobdispatcher.UbaJobSubmitter  # 自定义的cron，可以通过admin-web修改作业信息  mySubmitCronExpression=0 53/10 2-3 \* \* ? |

### Lts-admin界面手动添加作业

### Sql语句插入作业

|  |
| --- |
| INSERT INTO `lts\_cron\_job\_queue` VALUES ('2', '464DED22E434BF78FE0E9AF5C5B869BF', '100', '0', 'gaq\_start\_infos', '1456830045787', '1456830045787', 'uba\_taskTracker', 'uba\_taskTracker', '{\"product\":\"gaq\",\"type\":\"start\_infos\",\"size\":\"300\"}', '0', null, '0', '0 7 3-4 \* \* ?', '1456859220000'); |

Lts\_executable\_job\_queue 也要插入一条

# 配置中心

参见Disconf -环境搭建.docx

# Mq rpc

Zbus

# 消息推送

# 缓存