# Spring+Quartz定时任务最简集群版

## pom.xml

这个工程是作为我的Dubbo工程中的一部分，所以里面有一些依赖的其他Dubbo公共工程，大家使用的时候按照实际情况进行修改。

我把定时任务打成war包，大家也可以按照实际需求打成jar。

|  |
| --- |
| <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  <modelVersion>4.0.0</modelVersion>  <parent>  <groupId>org.leo.common</groupId>  <artifactId>common-parent</artifactId>  <version>1.0-SNAPSHOT</version>  <relativePath>../common-parent</relativePath>  </parent>  <description>用户服务的定时任务</description>  <groupId>org.leo.quartz</groupId>  <artifactId>quartz-user</artifactId>  <version>${quartz-user.version}</version>  <packaging>war</packaging>  <name>quartz-user</name>  <url>http://maven.apache.org</url>  <properties>  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  </properties>  <dependencies>  <dependency>  <groupId>org.leo.common</groupId>  <artifactId>common-config</artifactId>  <version>${common-config.version}</version>  </dependency>  <dependency>  <groupId>junit</groupId>  <artifactId>junit</artifactId>  <scope>test</scope>  </dependency>  <!-- 日志适配器 -->  <dependency>  <groupId>org.apache.logging.log4j</groupId>  <artifactId>log4j-slf4j-impl</artifactId>  <scope>test</scope>  </dependency>  <!-- 日志实现 -->  <dependency>  <groupId>org.apache.logging.log4j</groupId>  <artifactId>log4j-core</artifactId>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-core</artifactId>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-context-support</artifactId>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-web</artifactId>  </dependency>  <!-- Spring事务的依赖 -->  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-jdbc</artifactId>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-tx</artifactId>  </dependency>  <!-- 定时任务框架Quartz的依赖 -->  <dependency>  <groupId>org.quartz-scheduler</groupId>  <artifactId>quartz</artifactId>  </dependency>  <!-- MySQL -->  <dependency>  <groupId>mysql</groupId>  <artifactId>mysql-connector-java</artifactId>  </dependency>  <!-- 数据库连接池 -->  <dependency>  <groupId>com.alibaba</groupId>  <artifactId>druid</artifactId>  </dependency>  </dependencies>  </project> |

## web.xml

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <web-app version=*"2.5"* xmlns=*"http://java.sun.com/xml/ns/javaee"*  xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance*  *http://www.springmodules.org/schema/cache/springmodules-cache.xsd*  *http://www.springmodules.org/schema/cache/springmodules-ehcache.xsd"*  xsi:schemaLocation=*"http://java.sun.com/xml/ns/javaee*  *http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd*  *"*>  <display-name>用户定时任务</display-name>  <!-- 加载spring容器 -->  <context-param>  <param-name>contextConfigLocation</param-name>  <param-value>classpath:spring/applicationContext.xml</param-value>  </context-param>  <listener>  <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>  </listener>  <filter>  <filter-name>CharacterEncodingFilter</filter-name>  <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>  <init-param>  <param-name>encoding</param-name>  <param-value>UTF-8</param-value>  </init-param>  </filter>  <filter-mapping>  <filter-name>CharacterEncodingFilter</filter-name>  <url-pattern>/</url-pattern>  </filter-mapping>  </web-app> |

## applicationContext.xml

Quartz集群版需要数据库的支持，各种数据库的建表语句在Quartz官网下载的tar包里。

这里数据库连接的配置就不写了。

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <beans xmlns=*"http://www.springframework.org/schema/beans"*  xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:jdbc=*"http://www.springframework.org/schema/jdbc"*  xmlns:context=*"http://www.springframework.org/schema/context"*  xsi:schemaLocation=*"*  *http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd*  *http://www.springframework.org/schema/jdbc http://www.springframework.org/schema/jdbc/spring-jdbc.xsd*  *http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd*  *"*>  <context:component-scan base-package=*"org.leo.ssm"* />  <!-- 属性文件读入 -->  <context:property-placeholder  location=*"classpath:db.properties"* />  <!-- 基于Druid数据库链接池的数据源配置 -->  <bean id=*"dataSource"* class=*"com.alibaba.druid.pool.DruidDataSource"*  init-method=*"init"* destroy-method=*"close"*>  <!-- 基本属性driverClassName、 url、user、password -->  <property name=*"driverClassName"* value=*"com.mysql.jdbc.Driver"* />  <property name=*"url"* value=*"${jdbc.url}"* />  <property name=*"username"* value=*"${jdbc.username}"* />  <property name=*"password"* value=*"${jdbc.password}"* />  <!-- 配置初始化大小、最小、最大 -->  <!-- 通常来说，只需要修改initialSize、minIdle、maxActive -->  <property name=*"initialSize"* value=*"2"* />  <property name=*"minIdle"* value=*"2"* />  <property name=*"maxActive"* value=*"30"* />  <property name=*"testWhileIdle"* value=*"false"* />  <!-- 配置获取连接等待超时的时间 -->  <property name=*"maxWait"* value=*"5000"* />  <!-- 配置一个连接在池中最小生存的时间，单位是毫秒 -->  <property name=*"minEvictableIdleTimeMillis"* value=*"30000"* />  <!-- 配置间隔多久才进行一次检测，检测需要关闭的空闲连接，单位是毫秒 -->  <property name=*"timeBetweenEvictionRunsMillis"* value=*"60000"* />  <!-- 解密密码必须要配置的项 -->  <property name=*"filters"* value=*"config"* />  <property name=*"connectionProperties"* value=*"config.decrypt=false"* />  </bean>  <import resource=*"applicationContext-quartz.xml"* />  </beans> |

## applicationContext-quartz.xml

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <beans xmlns=*"http://www.springframework.org/schema/beans"*  xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*  xsi:schemaLocation=*"http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd"*>  <bean name=*"quartzScheduler"*  class=*"org.springframework.scheduling.quartz.SchedulerFactoryBean"*>  <property name=*"dataSource"*>  <ref bean=*"dataSource"* />  </property>  <property name=*"applicationContextSchedulerContextKey"* value=*"applicationContextKey"* />  <property name=*"configLocation"* value=*"classpath:quartz.properties"* />  <property name=*"triggers"*>  <list>  <ref bean=*"trigger1"* />  <ref bean=*"trigger2"* />  </list>  </property>  </bean>  <bean id=*"jobDetail1"* class=*"org.springframework.scheduling.quartz.JobDetailFactoryBean"*>  <property name=*"jobClass"*>  <value>org.leo.ssm.quartz.user.TestUserQuartzA</value>  </property>  <property name=*"durability"* value=*"true"* />  <property name=*"requestsRecovery"* value=*"true"* />  </bean>  <bean id=*"trigger1"* class=*"org.springframework.scheduling.quartz.CronTriggerFactoryBean"*>  <property name=*"jobDetail"* ref=*"jobDetail1"* />  <property name=*"cronExpression"* value=*"0/10 \* \* ? \* \* \*"* />  </bean>  <bean id=*"jobDetail2"* class=*"org.springframework.scheduling.quartz.JobDetailFactoryBean"*>  <property name=*"jobClass"*>  <value>org.leo.ssm.quartz.user.TestUserQuartzB</value>  </property>  <property name=*"durability"* value=*"true"* />  <property name=*"requestsRecovery"* value=*"true"* />  </bean>  <bean id=*"trigger2"* class=*"org.springframework.scheduling.quartz.CronTriggerFactoryBean"*>  <property name=*"jobDetail"* ref=*"jobDetail2"* />  <property name=*"cronExpression"* value=*"0/20 \* \* ? \* \* \*"* />  </bean>  </beans> |

## 5、定时任务类

|  |
| --- |
| @PersistJobDataAfterExecution  @DisallowConcurrentExecution // 不允许并发执行  **public** **class** TestUserQuartzA **extends** QuartzJobBean {  @Override  **protected** **void** executeInternal(JobExecutionContext arg0) **throws** JobExecutionException {  // TestUserQuartzB代码相同。  //因为是集群版，为了方便查看Log，所以在编译的时候，放在主Tomcat的war打印的是M-，从Tomcat是S-。  System.***out***.println("M-TestUserQuartzA执行于:" + System.*currentTimeMillis*());  }  } |

这个实现是最简易版的，实际项目中，肯定是要连接数据库（我的工程是要连Dubbo服务的），所以其他配置大家自己往里添加即可。