Spring task使用详解

# Spring task简介

spring通过接口TaskExecutor和TaskScheduler这两个接口的方式为异步定时任务提供了一种抽象。这就意味着spring容许你使用其他的定时任务框架，当然spring自身也提供了一种定时任务的实现：spring task。spring task支持线程池，可以高效处理许多不同的定时任务。同时，spring还支持使用Java自带的Timer定时器和Quartz定时框架。

## TaskExecutor和TaskScheduler

**TaskExecutor**：spring task的第一个抽象，它很自然让人联想到jdk中concurrent包下的Executor，实际上TaskExecutor就是为区别于Executor才引入的，而引入TaskExecutor的目的就是为定时任务的执行提供线程池的支持，那么，问题来了，为什么spring不直接使用jdk自带的Executor呢？TaskExecutor源码如下？

|  |
| --- |
| public interface TaskExecutor extends Executor {  void execute(Runnable var1);  }  那么，答案很显然，TaskExecutor提供的线程池支持也是基于jdk自带的Executor的。用法于Executor没有什么不同。 |

**TaskScheduler**：是spring task的第二个抽象，那么从字面的意义看，Task  
Scheduler就是为了提供定时任务的支持。TaskScheduler需要传入一个Run  
nable的任务做为参数，并指定需要周期执行的时间或者触发器，这样Runna  
ble任务就可以周期性执行了。传入时间很好理解，有意思的是传入一个触发器（Trigger）的情况，因为这里需要使用cron表达式去触发一个定时任务，所以有必要先了解下cron表达式的使用。在spring 4.x中**已经不支持7个参数的cronin表达式了**，要求必须是6个参数（具体哪个参数后面会说）。**cron表达式的格式如下：**

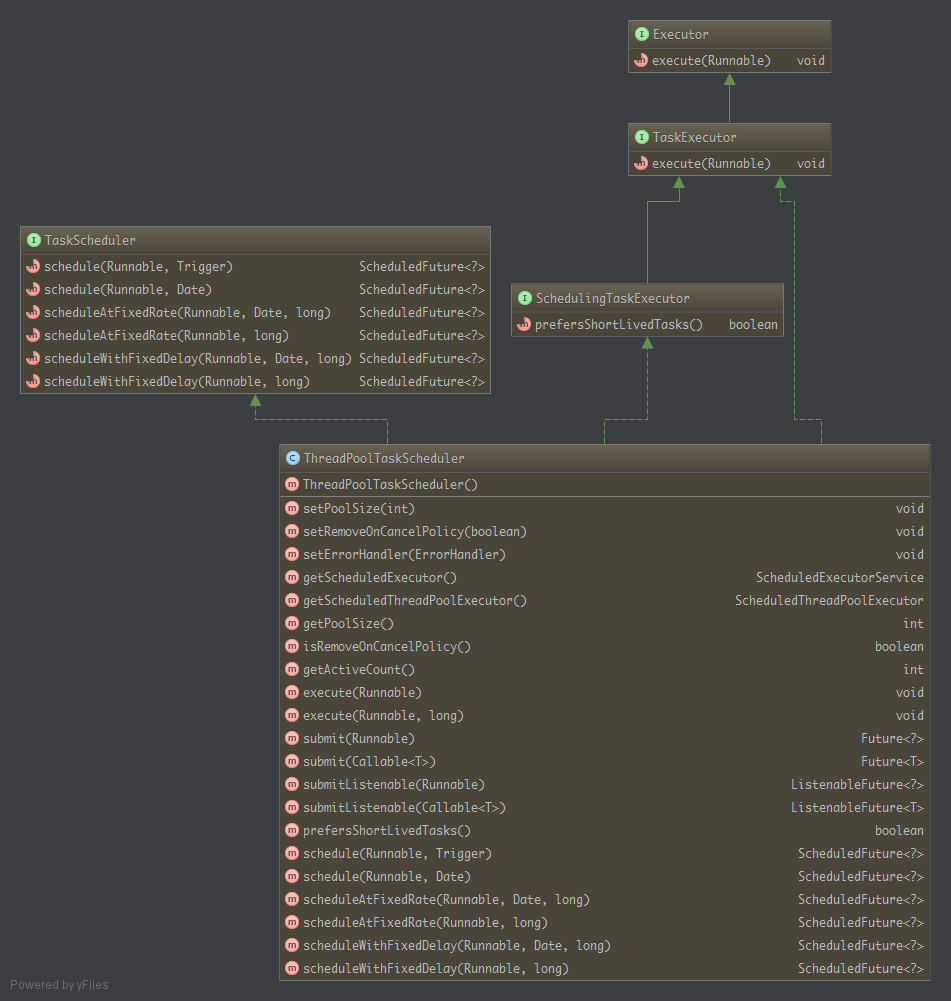
|  |
| --- |
| {秒} {分} {时} {日期（具体哪天）} {月} {星期}  说明：   * **秒**：必填项，允许的值范围是0-59，支持的特殊符号包括 , - \* /，,表示特定的某一秒才会触发任务，-表示一段时间内会触发任务，\*表示每一秒都会触发，/表示从哪一个时刻开始，每隔多长时间触发一次任务。 * **分**：必填项，允许的值范围是0-59，支持的特殊符号和秒一样，含义类推 * **时**：必填项，允许的值范围是0-23，支持的特殊符号和秒一样，含义类推 * **日期**：必填项，允许的值范围是1-31，支持的特殊符号相比秒多了?，表示与{星期}互斥，即意味着若明确指定{星期}触发，则表示{日期}无意义，以免引起冲突和混乱。 * **月**：必填项，允许的值范围是1-12（JAN-DEC），支持的特殊符号与秒一样，含义类推 * **星期**：必填项，允许值范围是1~7 (SUN-SAT),1代表星期天（一星期的第一天），以此类推，7代表星期六，支持的符号相比秒多了?，表达的含义是与{日期}互斥，即意味着若明确指定{日期}触发，则表示{星期}无意义。 |

比如下面这个cron表达式：30 \* \* \* \* ? //每半分钟触发一次任务

spring提供了一个CronTrigger，通过传入一个Runnable任务和CronTrig  
ger，就可以使用cron表达式去指定定时任务了，是不是非常方面。实际上，在工程实践上，cron表达式也是使用很多的。实际上，是执行了下面的代码：

|  |
| --- |
| scheduler.schedule(task, new CronTrigger("30 \* \* \* \* ?")); |

TaskScheduler抽象的好处是让需要执行定时任务的代码不需要指定特定的定时框架（比如Timer和Quartz）。TaskScheduler的更简单的实现是Thread  
PoolTaskScheduler,它实际上代理一个jdk中的SchedulingTaskExecutor，并且也实现了TaskExecutor接口，所以需要经常执行定时任务的场景可以使用这个实现我们再来看一下TaskExecutor和TaskScheduler的类继承关系：



# Spring task使用

## maven依赖

|  |
| --- |
| <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-context</artifactId>  <version>4.3.1.RELEASE</version>  </dependency> |

## spring task配置

|  |
| --- |
| <?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:context=*"http://www.springframework.org/schema/context"*  xmlns:task=*"http://www.springframework.org/schema/task"*  xsi:schemaLocation=*"*  *http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-4.1.xsd*  *http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.1.xsd*  *http://www.springframework.org/schema/task http://www.springframework.org/schema/task/spring-task-4.1.xsd"*>  <!-- 配置注解扫描 -->  <context:component-scan base-package=*"springtask"*/>  <task:scheduler id=*"taskScheduler"* pool-size=*"100"* />  <task:scheduled-tasks scheduler=*"taskScheduler"*>  <!-- 每半分钟触发任务 -->  <task:scheduled ref=*"app"* method=*"execute1"* cron=*"30 \* \* \* \* ?"*/>  <!-- 每小时的10分30秒触发任务 -->  <task:scheduled ref=*"app"* method=*"execute2"* cron=*"30 10 \* \* \* ?"*/>  <!-- 每天1点10分30秒触发任务 -->  <task:scheduled ref=*"app"* method=*"execute3"* cron=*"30 10 1 \* \* ?"*/>  <!-- 每月20号的1点10分30秒触发任务 -->  <task:scheduled ref=*"app"* method=*"execute4"* cron=*"30 10 1 20 \* ?"*/>  <!-- 每年10月20号的1点10分30秒触发任务 -->  <task:scheduled ref=*"app"* method=*"execute5"* cron=*"30 10 1 20 10 ?"*/>  <!-- 每15秒、30秒、45秒时触发任务 -->  <task:scheduled ref=*"app"* method=*"execute6"* cron=*"15,30,45 \* \* \* \* ?"*/>  <!-- 15秒到45秒每隔1秒触发任务 -->  <task:scheduled ref=*"app"* method=*"execute7"* cron=*"15-45 \* \* \* \* ?"*/>  <!-- 每分钟的每15秒时任务任务，每隔5秒触发一次 -->  <task:scheduled ref=*"app"* method=*"execute8"* cron=*"15/5 \* \* \* \* ?"*/>  <!-- 每分钟的15到30秒之间开始触发，每隔5秒触发一次 -->  <task:scheduled ref=*"app"* method=*"execute9"* cron=*"15-30/5 \* \* \* \* ?"*/>  <!-- 每小时的0分0秒开始触发，每隔3分钟触发一次 -->  <task:scheduled ref=*"app"* method=*"execute10"* cron=*"0 0/3 \* \* \* ?"*/>  <!-- 星期一到星期五的10点15分0秒触发任务 -->  <task:scheduled ref=*"app"* method=*"execute11"* cron=*"0 15 10 ? \* MON-FRI"*/>  </task:scheduled-tasks>  </beans> |
| #定时任务  @Component  **public** **class** App {  **public** **void** execute1() {  System.***out***.printf("Task: %s, Current time: %s\n", 1, LocalDateTime.*now*());  }  **public** **void** execute2() {  System.***out***.printf("Task: %s, Current time: %s\n", 2, LocalDateTime.*now*());  }  **public** **void** execute3() {  System.***out***.printf("Task: %s, Current time: %s\n", 3, LocalDateTime.*now*());  }  **public** **void** execute4() {  System.***out***.printf("Task: %s, Current time: %s\n", 4, LocalDateTime.*now*());  }  **public** **void** execute5() {  System.***out***.printf("Task: %s, Current time: %s\n", 5, LocalDateTime.*now*());  }  **public** **void** execute6() {  System.***out***.printf("Task: %s, Current time: %s\n", 6, LocalDateTime.*now*());  }  **public** **void** execute7() {  System.***out***.printf("Task: %s, Current time: %s\n", 7, LocalDateTime.*now*());  }  **public** **void** execute8() {  System.***out***.printf("Task: %s, Current time: %s\n", 8, LocalDateTime.*now*());  }  **public** **void** execute9() {  System.***out***.printf("Task: %s, Current time: %s\n", 9, LocalDateTime.*now*());  }  **public** **void** execute10() {  System.***out***.printf("Task: %s, Current time: %s\n", 10, LocalDateTime.*now*());  }  **public** **void** execute11() {  System.***out***.printf("Task: %s, Current time: %s\n", 11, LocalDateTime.*now*());  }  } |

## Spring task测试

|  |
| --- |
| **public** **static** **void** main(String[] args) {  ApplicationContext context = **new** ClassPathXmlApplicationContext("classpath:/spring-task.xml");  } |

## 注解实现配置

|  |
| --- |
| <task:annotation-driven /> --->配置到spring中  @Scheduled(cron="0/1 \* \* \* \* ?") --->配置到定时任务的方法上面 |