<http://blog.csdn.net/qq_33556185/article/details/51852537>

在springMVC里使用spring的定时任务非常的简单，如下：

**（一）在xml里加入task的命名空间**

**[html]** [view plain](http://blog.csdn.net/qq_33556185/article/details/51852537) [copy](http://blog.csdn.net/qq_33556185/article/details/51852537)

1. xmlns:task="http://www.springframework.org/schema/task"
2. http://www.springframework.org/schema/task http://www.springframework.org/schema/task/spring-task-4.1.xsd

**（二）启用注解驱动的定时任务**

**[html]** [view plain](http://blog.csdn.net/qq_33556185/article/details/51852537) [copy](http://blog.csdn.net/qq_33556185/article/details/51852537)

1. **<task:annotation-driven** scheduler="myScheduler"**/>**

**（三）配置定时任务的线程池**

推荐配置线程池，若不配置多任务下会有问题。后面会详细说明单线程的问题。

**[html]** [view plain](http://blog.csdn.net/qq_33556185/article/details/51852537) [copy](http://blog.csdn.net/qq_33556185/article/details/51852537)

1. **<task:scheduler** id="myScheduler" pool-size="5"**/>**

**（四）写我们的定时任务**

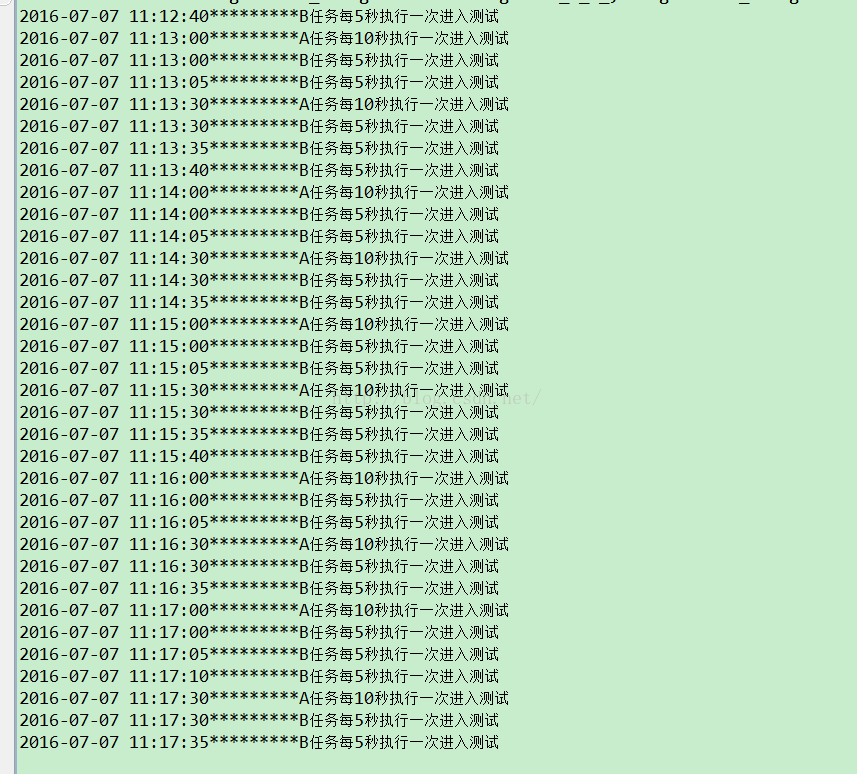
@Scheduled注解为定时任务，cron表达式里写执行的时机

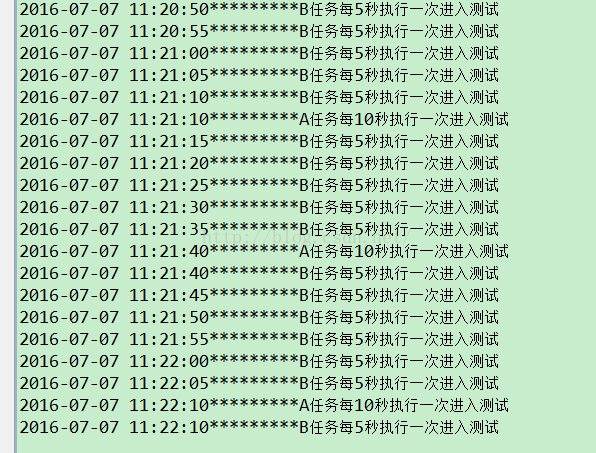
**[java]** [view plain](http://blog.csdn.net/qq_33556185/article/details/51852537) [copy](http://blog.csdn.net/qq_33556185/article/details/51852537)

1. **package** com.mvc.task.impl;
2. **import** java.text.DateFormat;
3. **import** java.text.SimpleDateFormat;
4. **import** java.util.concurrent.TimeUnit;
5. **import** org.joda.time.DateTime;
6. **import** org.springframework.scheduling.annotation.Scheduled;
7. **import** org.springframework.stereotype.Component;
8. **import** com.mvc.task.IATask;
9. @Component
10. **public** **class** ATask **implements** IATask{
11. @Scheduled(cron="0/10 \* \*  \* \* ? ")   //每10秒执行一次
12. @Override
13. **public** **void** aTask(){
14. **try** {
15. TimeUnit.SECONDS.sleep(20);
16. } **catch** (InterruptedException e) {
17. e.printStackTrace();
18. }
19. DateFormat sdf = **new** SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
20. System.out.println(sdf.format(DateTime.now().toDate())+"\*\*\*\*\*\*\*\*\*A任务每10秒执行一次进入测试");
21. }
22. }

**[java]** [view plain](http://blog.csdn.net/qq_33556185/article/details/51852537) [copy](http://blog.csdn.net/qq_33556185/article/details/51852537)

1. **package** com.mvc.task.impl;
2. **import** java.text.DateFormat;
3. **import** java.text.SimpleDateFormat;
4. **import** org.joda.time.DateTime;
5. **import** org.springframework.scheduling.annotation.Scheduled;
6. **import** org.springframework.stereotype.Component;
7. **import** com.mvc.task.IBTask;
8. @Component
9. **public** **class** BTask **implements** IBTask{
10. @Scheduled(cron="0/5 \* \*  \* \* ? ")   //每5秒执行一次
11. @Override
12. **public** **void** bTask(){
13. DateFormat sdf = **new** SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
14. System.out.println(sdf.format(DateTime.now().toDate())+"\*\*\*\*\*\*\*\*\*B任务每5秒执行一次进入测试");
15. }
16. }

spring的定时任务默认是单线程**，**多个任务执行起来时间会有问题（B任务会因为A任务执行起来需要20S而被延后20S执行），如下图所示： ****当我们配置了线程池后再来看结果（多线程下，B任务再也不会因为A任务执行起来要20S而被延后了）：

****

**cron表达式详解：**

一个cron表达式有至少6个（也可能7个）有空格分隔的时间元素。

       按顺序依次为  
      1  秒（0~59）  
      2  分钟（0~59）  
      3 小时（0~23）  
      4  天（0~31）  
      5 月（0~11）  
      6  星期（1~7 1=SUN 或 SUN，MON，TUE，WED，THU，FRI，SAT）  
      7.年份（1970－2099）  
      其中每个元素可以是一个值(如6),一个连续区间(9-12),一个间隔时间(8-18/4)(/表示每隔4小时),一个列表(1,3,5),通配符。由于"月份中的日期"和"星期中的日期"这两个元素互斥的,必须要对其中一个设置?.  
       0 0 10,14,16 \* \* ? 每天上午10点，下午2点，4点  
       0 0/30 9-17 \* \* ?   朝九晚五工作时间内每半小时  
       0 0 12 ? \* WED 表示每个星期三中午12点  
       "0 0 12 \* \* ?" 每天中午12点触发   
       "0 15 10 ? \* \*" 每天上午10:15触发   
       "0 15 10 \* \* ?" 每天上午10:15触发   
       "0 15 10 \* \* ? \*" 每天上午10:15触发   
       "0 15 10 \* \* ? 2005" 2005年的每天上午10:15触发   
       "0 \* 14 \* \* ?" 在每天下午2点到下午2:59期间的每1分钟触发   
       "0 0/5 14 \* \* ?" 在每天下午2点到下午2:55期间的每5分钟触发   
       "0 0/5 14,18 \* \* ?" 在每天下午2点到2:55期间和下午6点到6:55期间的每5分钟触发   
       "0 0-5 14 \* \* ?" 在每天下午2点到下午2:05期间的每1分钟触发   
       "0 10,44 14 ? 3 WED" 每年三月的星期三的下午2:10和2:44触发   
       "0 15 10 ? \* MON-FRI" 周一至周五的上午10:15触发   
       "0 15 10 15 \* ?" 每月15日上午10:15触发   
       "0 15 10 L \* ?" 每月最后一日的上午10:15触发   
       "0 15 10 ? \* 6L" 每月的最后一个星期五上午10:15触发   
       "0 15 10 ? \* 6L 2002-2005" 2002年至2005年的每月的最后一个星期五上午10:15触发   
       "0 15 10 ? \* 6#3" 每月的第三个星期五上午10:15触发   
       有些子表达式能包含一些范围或列表  
       例如：子表达式（天（星期））可以为 “MON-FRI”，“MON，WED，FRI”，“MON-WED,SAT”  
       “\*”字符代表所有可能的值  
       “/”字符用来指定数值的增量  
       例如：在子表达式（分钟）里的“0/15”表示从第0分钟开始，每15分钟  
                在子表达式（分钟）里的“3/20”表示从第3分钟开始，每20分钟（它和“3，23，43”）的含义一样  
       “？”字符仅被用于天（月）和天（星期）两个子表达式，表示不指定值  
       当2个子表达式其中之一被指定了值以后，为了避免冲突，需要将另一个子表达式的值设为“？”  
       “L” 字符仅被用于天（月）和天（星期）两个子表达式，它是单词“last”的缩写  
       如果在“L”前有具体的内容，它就具有其他的含义了。例如：“6L”表示这个月的倒数第６天  
       注意：在使用“L”参数时，不要指定列表或范围，因为这会导致问题  
       W 字符代表着平日(Mon-Fri)，并且仅能用于日域中。它用来指定离指定日的最近的一个平日。大部分的商业处理都是基于工作周的，所以 W 字符可能是非常重要的。  
       例如，日域中的 15W 意味着 "离该月15号的最近一个平日。" 假如15号是星期六，那么 trigger 会在14号(星期五)触发，因为星期四比星期一离15号更近。  
       C：代表“Calendar”的意思。它的意思是计划所关联的日期，如果日期没有被关联，则相当于日历中所有日期。例如5C在日期字段中就相当于日历5日以后的第一天。1C在星期字段中相当于星期日后的第一天。  
       字段   允许值   允许的特殊字符  
       秒           0-59           , - \* /  
       分           0-59           , - \* /  
       小时           0-23           , - \* /  
       日期           1-31           , - \* ? / L W C  
       月份           1-12 或者 JAN-DEC           , - \* /  
       星期           1-7 或者 SUN-SAT           , - \* ? / L C #  
       年（可选）           留空, 1970-2099           , - \* /

版权声明：本文为博主原创文章，未经博主允许不得转载。