

【SpringBoot】@Scheduled 定時執行

在Spring Boot中，我可以使用注解@Scheduled来定义定时执行方法。该注解可用于方法上，需要定期执行的方法，并提供执行计划。

@Scheduled注解支持以下属性：

1. fixedRate：以毫秒为单位的执行速率。
2. fixedDelay：以毫秒为单位的执行延迟，即上一次执行结束后的延迟时间。
3. initialDelay：第一次执行的延迟时间，以毫秒为单位。
4. cron：CRON表达式，用于更高的定时计划。

下面是一个示例，示例演示如何使用@Scheduled注解在Spring Boot中定期执行方法：

```
import org.springframework.scheduling.annotation.Scheduled;
import org.springframework.stereotype.Component;

@Component
public class MyTask {

    @Scheduled(fixedRate = 5000)
    public void printMessage() {
        System.out.println("Hello, world!");
    }
}
```

在上述示例中，我定义了一个名为MyTask的组件，其中包含一个使用@Scheduled注解的printMessage方法。此方法在每隔5秒钟输出一条消息"Hello, world!"。

我可以使用fixedDelay属性来延迟执行。例如，如果我设置fixedDelay为10000，那么方法在上一次执行结束后等待10秒才能开始下一次执行。

@Scheduled注解支持CRON表达式，使得我可以更高地安排任务。例如，如果我想要在每天早上6点执行任务，我可以使用以下CRON表达式：

```
0 0 6 * * *
```

在此示例中，我使用六个星号指定秒，分，时，日，月和星期几。因此，此表达式在每天早上6点执行任务。

註解停用

如果你想停用@Scheduled注解的方法，你可以简单地注释掉它，或者将其禁用，以便稍后重新启用。

要注释掉方法，可以注释掉：

```
// @Scheduled(fixedRate = 5000)
public void printMessage() {
    System.out.println("Hello, world!");
}
```

要禁用方法，您可以使用@Scheduled注解的enabled属性：

```
// @Scheduled(fixedRate = 5000)
public void printMessage() {
    System.out.println("Hello, world!");
}
```

在例子中，我使用@Scheduled注解的enabled属性来禁用printMessage方法。要重新启用方法，只需将enabled属性设置为true即可。

使用命令停用

是的，您可以使用Spring Boot Actuator的端点动态启用/禁用定时任务。Spring Boot Actuator是Spring Boot的一个附加模块，提供了很多有用的端点，可以在应用程序运行期间视和管理应用程序。

您需要在项目中添加Actuator依赖：

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
```

然后，您可以使用`/actuator/scheduledtasks`端点查看和管理定时任务。要停用定时任务，您可以使用端点取消划定的任务。例如：

```
@Scheduled(fixedRate = 5000, name = "myTask")
public void printMessage() {
    System.out.println("Hello, world!");
}
```

```
$ curl -X POST http://localhost:8080/actuator/scheduledtasks/taskName
```

取消划名`taskName`的任务。要重新启用任务，您可以使用端点重新启用任务：

```
$ curl -X DELETE http://localhost:8080/actuator/scheduledtasks/taskName
```

假设您的Spring Boot应用程序正在运行，并且您已经添加了Actuator依赖，那么您可以使用以下curl命令向`/actuator/scheduledtasks`端点发出HTTP GET请求获取前应用程序中所有定时任务的列表：

```
curl http://localhost:8080/actuator/scheduledtasks
```

返回一个JSON格式的响应，其中包含有每个定时任务的信息，例如任务名、任务是否启用、任务的CRON表达式等。您可以使用响应查看应用程序中所有的定时任务信息。

```
{
  "cron": {
    "myScheduledTask": {
      "scheduled": true,
      "expression": "0/30 * * * * ?"
    }
  },
  "fixedDelay": {},
  "fixedRate": {}
}
```

🕒 修订版本 #1

★ 由 treeman 建立於 5 🕒 2023 10:19:35

✍ 由 treeman 更新於 5 🕒 2023 10:38:31