# 2022 7.12

## 工作说明

实现了大作业`1.功能要求`中的第1，2点功能。

## 详细步骤

### 1.创建并初始化Spring boot项目

1. 使用idea插件`Spring boot initializer`初始化一个名为`cloud-native-proj`的springboot项目。
2. 在`src/main/resource/application.properties`文件下修改web服务端口：

```
server.port=8080
```

3. 新建`src/main/java/com/example/cloud_native_proj/controller`文件夹，在该文件夹下新建`UserController.java`文件，并自定义`UserController`类，返回基本的`json`信息。（返回json信息需要导入`alibaba fastjson`依赖）

```
<!--json工具-->
<dependency>
    <groupId>com.alibaba</groupId>
    <artifactId>fastjson</artifactId>
    <version>1.2.15</version>
</dependency>
```

导入*json*依赖

```
public class UserController {
    //@ResponseBody
    @RequestMapping("/hello")
    //每秒并发量，设置为10
    @CurrentLimiter(QPS = 10)
    public Object sayHello() {
        JSONObject jsonObject=new JSONObject();

        jsonObject.put("code",200);
        jsonObject.put("group","1");
        jsonObject.put("member","Qinyu Chen,Jifeng Duan,Huiyuan Yan");
        jsonObject.put("msg","Welcome!");
        return jsonObject.toString();
    }
```

*UserController*类

## 2.实现接口限流功能

采用`current limiting`工具实现了接口限流工作，但还没有实现多pod统一限流。

1. 引入依赖

```
<!--接口限流，采用Current Limiting 工具进行-->
<dependency>
    <groupId>cn.yueshutong</groupId>
    <artifactId>spring-boot-starter-current-limiting</artifactId>
    <version>0.0.8.RELEASE</version>
</dependency>
```

2. 在`src`根文件夹下新建`application.yaml`配置文件并写入流量控制的相关配置内容：

```
current:
  limiting:
    #开启全局限流
    enabled: false
    #开启注解限流,可使注解失效
    part-enabled: true
    #每秒并发量 这里的qps是全局限流开启的时候的值,如果使用注解在注解里设置QPS值
    qps: 100
    #开启快速失败,可切换为阻塞
    fail-fast: true
    #系统启动保护时间为0
    initial-delay: 0
```

3. 在`UserController`类下添加注解：

```
public class UserController {
    //@ResponseBody
    @RequestMapping("/hello")
    //每秒并发量，设置为10
    @CurrentLimiter(QPS = 10)
    public Object sayHello() {
```

为测试方便，每秒最大限流为10次（作业最终提交时可修改成100次）。

4. 新建`src/main/java/com/example/cloud_native_proj/co/FC`文件夹(Flow Control)，在该文件夹下新建`MyCurrentLimitHandler.java`文件，并自定义处理类处理超时函数。

```java
@Component
public class MyCurrentLimitHandler  implements CurrentAspectHandler {
    @Override
    public Object around(ProceedingJoinPoint pjp, CurrentLimiter rateLimiter)  {

        //限流的返回数据可以自己根据需求场景设计

        JSONObject jsonObject=new JSONObject();

        jsonObject.put("code",429);
        jsonObject.put("msg","Too Too many requests");
        return jsonObject.toString();
        //return "Too many!!!";
    }
}
```

# 功能演示

1. 编译运行

   在项目文件夹下运行命令：

   ```
   mvn clean install package '-Dmaven.test.skip=true'
   ```

   ```
   java -jar target/cloud_native_proj-0.0.1-SNAPSHOT.jar
   ```

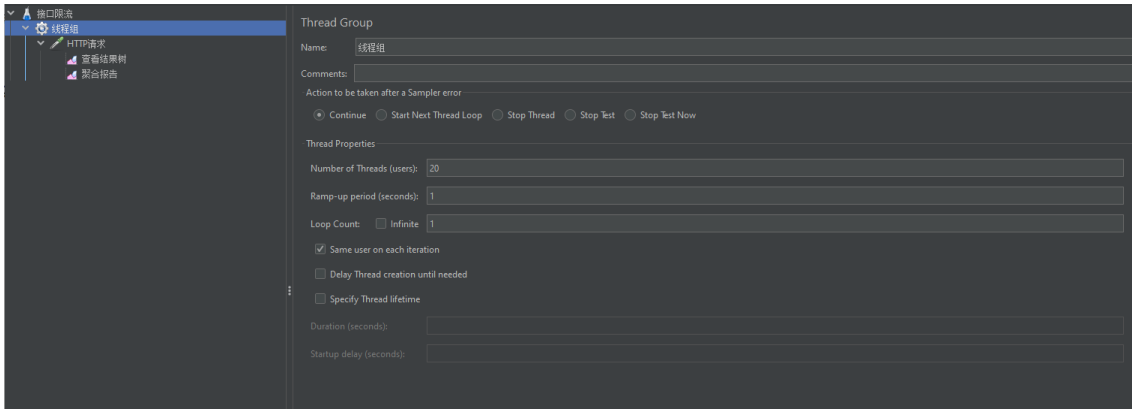
   构建并运行项目。

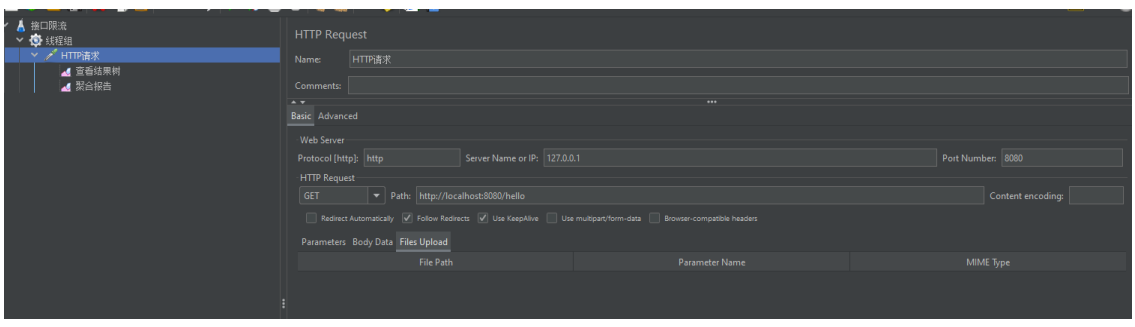
2. 打开浏览器输入网址：`http://localhost:8080/hello`（注意mapping有`/hello`）查看内容：

   {"msg":"Welcome!","code":200,"member":"Qinyu Chen,Jifeng Duan,Huiyuan Yan","group":"1"}

3. 用`Apache Jmeter`工具对接口进行压力测试。

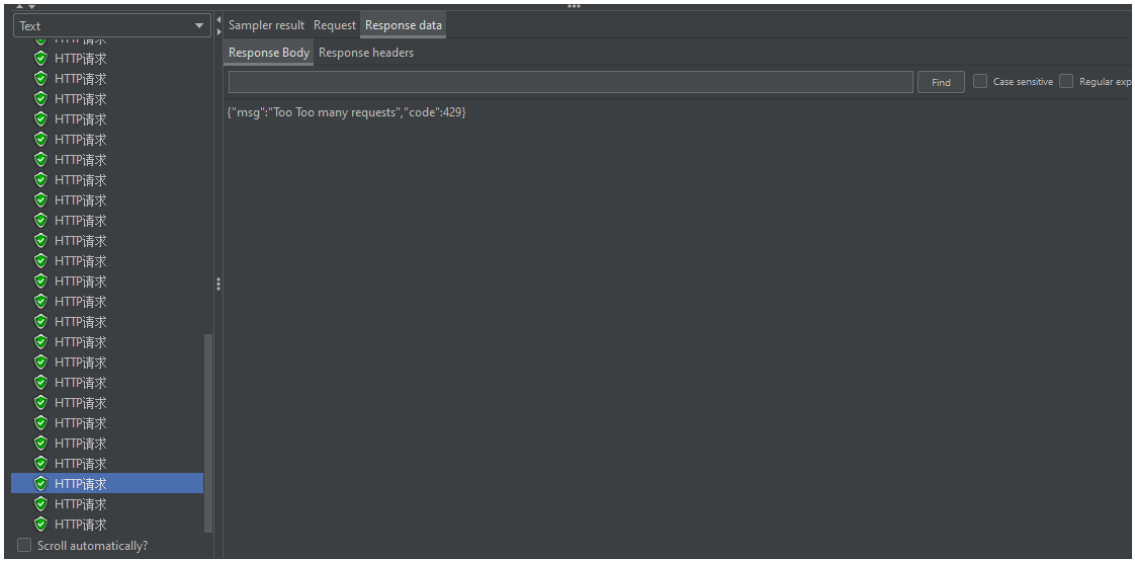
   配置如下：



会生成jmx文件。

然后点击右上角绿色箭头运行，在结果树中查看情况。




**平均**一秒内每十次左右访问就会出现访问次数过多的返回值。