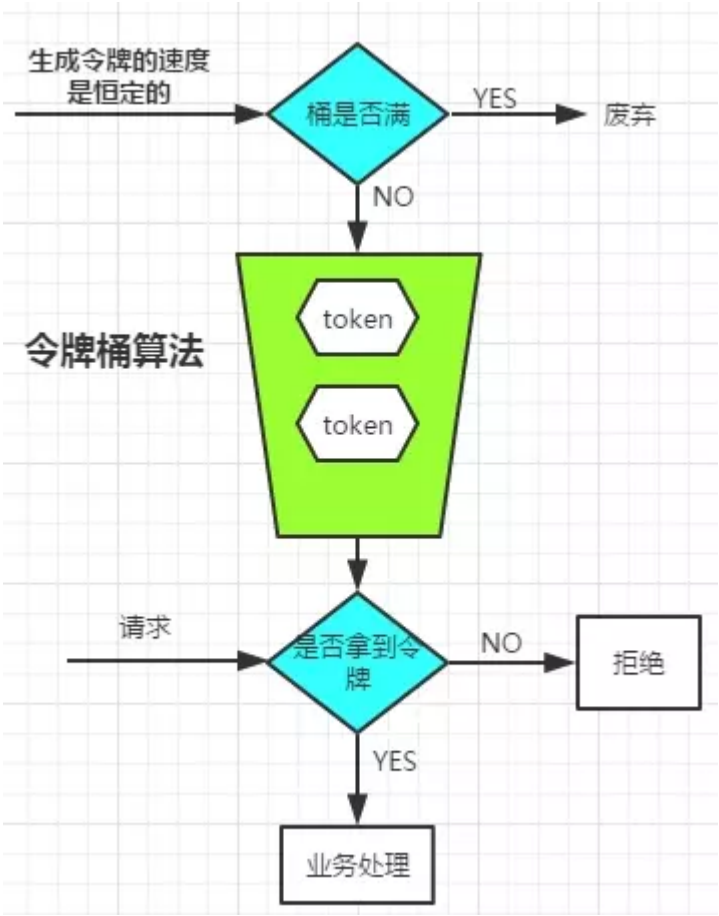
在springboot中使用Guava基于令牌桶实现限流

SpringBoot中... Lv2
2020年08月04日 21:21 · 阅读 2389

限流说详细了，名堂也多。这种算法那种算法，这种策略那种策略的。没有绝对的银弹。都要结合实际的场景来实现。最简单的，使用Google的Guava，几行代码。就可以优雅的对一个接口完成**限流**。

令牌桶算法



通俗的理解就是，有一个固定大小的水桶，水龙头一直按照一定的频率往里面滴水。水满了，就不滴了。客户端每次进行请求之前，都要先尝试从水桶里面起码取出“一滴水”，才能处理业务。因为桶的大小固定，水龙头滴水频率固定。从而也就保证了数据接口的访问流量。

谷歌的一个**工具库**，包含了大量的Java工具类，像hash算法，字符串处理，集合等等。。。

github.com/google/guav...

mava 复制代码

```
<!-- https://mvnrepository.com/artifact/com.google.guava/guava -->
<dependency>
    <groupId>com.google.guava</groupId>
    <artifactId>guava</artifactId>
    <version>29.0-jre</version>
</dependency>
```

速率限制器 RateLimiter

java 复制代码

```
/**
 * 创建一个限速器，每1秒，产生2.5个令牌
 */
RateLimiter rateLimiter = RateLimiter.create(2.5, 1, TimeUnit.SECONDS);

/**
 * 尝试获取1个令牌，如果没有，会阻塞当前线程。直到获取成功返回。
 * 返回值是，阻塞的秒数
 */
double waitSeconds = rateLimiter.acquire();

/**
 * 尝试获取1个令牌，不会阻塞当前线程。
 * 立即返回是否获取成功。
 */
boolean success = rateLimiter.tryAcquire();
```

好了，这就是核心代码。就3行。首先创建一个**限速器**，指定令牌的生产频率。 核心的方法就是2种，阻塞获取令牌，非阻塞获取令牌。代码也通俗易懂。

重载方法

不论是**阻塞获取令牌**还是**非阻塞获取令牌**，它们都有几个重载方法。一看也清楚，就是可以设置**获取令牌的数量**，以及阻塞的时间。

```
public boolean tryAcquire(Duration timeout)
public boolean tryAcquire(int permits)
public boolean tryAcquire(long timeout, TimeUnit unit)
public boolean tryAcquire(int permits, long timeout, TimeUnit unit)
public boolean tryAcquire(int permits, Duration timeout)
```

Controller，也就是被限速的接口

java 复制代码

```java
@RestController
@RequestMapping("/test")
public class TestController {

    @GetMapping
    public Object test () {
        return Collections.singletonMap("success", "true");
    }
}
```

RateLimiterInterceptor，负责实现限速逻辑

java 复制代码

```java
import java.nio.charset.StandardCharsets;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.http.MediaType;
import org.springframework.web.servlet.handler.HandlerInterceptorAdapter;

import com.google.common.util.concurrent.RateLimiter;

public class RateLimiterInterceptor extends HandlerInterceptorAdapter {

    private final RateLimiter rateLimiter;

    /**
     * 通过构造函数初始化限速器
     */
    public RateLimiterInterceptor(RateLimiter rateLimiter) {
```

```
    @Override
    public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler) throw

        if(this.rateLimiter.tryAcquire()) {
            /**
             * 成功获取到令牌
             */
            return true;
        }

        /**
         * 获取失败，直接响应“错误信息”
         * 也可以通过抛出异常，通过全全局异常处理器响应客户端
         */
        response.setCharacterEncoding(StandardCharsets.UTF_8.name());
        response.setContentType(MediaType.TEXT_PLAIN_VALUE);
        response.getWriter().write("服务器繁忙");
        return false;
    }
}
```

拦截器的配置

java 复制代码

```
import java.util.concurrent.TimeUnit;

import org.springframework.context.annotation.Configuration;
import org.springframework.web.servlet.config.annotation.InterceptorRegistry;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;

import com.google.common.util.concurrent.RateLimiter;

import io.springboot.jwt.web.interceptor.RateLimiterInterceptor;


@Configuration
public class WebMvcConfiguration implements WebMvcConfigurer {

    @Override
    public void addInterceptors(InterceptorRegistry registry) {
        /**
         * test接口，1秒钟生成1个令牌，也就是1秒中允许一个人访问
         */
        registry.addInterceptor(new RateLimiterInterceptor(RateLimiter.create(1, 1, TimeUnit.SECONDS)))
```

客户端演示限流效果

原文：springboot.io/t/topic/235...

分类： 阅读 标签： Spring Boot

评论

输入评论（Enter换行，Ctrl + Enter发送）

相关推荐

李子捌 6月前 Redis 后端

架构师如何讲解Redis限流——令牌桶限流

902 13 3

1180 10 2

不问心 9月前 Vue.js 前端

Vue打包压缩，利用gzip压缩代码

1578 10 评论

顽疾 8月前 后端 算法

限流算法-令牌桶算法 | 8月更文挑战

735 1 评论

Gaby 6月前 JavaScript 面试

连八股文都不懂还指望在前端混下去么

23.1w 6031 265

蛮三刀酱 2年前 架构

【秒杀系统】零基础上手秒杀系统（二）：令牌桶限流 + 再谈超卖

2643 27 4

小虎哥的技术博客 11月前 后端

SpringBoot JWT令牌保护、注册、登录、统一异常拦截、封装返回结果

819 6 评论

小平果118 3月前 Node.js

Guava令牌桶RateLimiter限流原理

644 3 评论

TEAVAMC 1年前 分布式

基于 Redis 和 Lua 实现分布式令牌桶限流

1117 6 2

逐九 1年前 Go

限流-令牌桶实现(go版本)

521 1 评论

7816 75 7

OBKoro1 4年前 Vue.js Webpack

vue-cli npm run build空白页的两个坑 webpack gzip文件压缩优化打包文件

1.2w 89 20

萤火架构 4月前 架构 Redis

使用Redis实现令牌桶算法

983 2 评论

万俊峰Kevin 3月前 Go 微服务

Go 分布式令牌桶限流 + 兜底策略

1788 7 评论

低调的码农 2年前 架构

高并发下漏洞桶限流设计方案 - Redis

2666 26 6

大帅老猿 11月前 前端 JavaScript

产品经理：你能不能用div给我画条龙?

11.1w 2948 585

萤火架构 4月前 架构

ASP.NET Core中使用令牌桶算法限流

450 点赞 评论

PHP进阶架构师 1年前 PHP

基于PHP+Redis令牌桶限流

1047 点赞 评论

坏丶毛病 1年前 Vue.js

vue打包时gzip压缩的两种方案

1496 2 5

1.5w 74 5