乐字节教育高级架构课程

正所谓“授人以鱼不如授人以渔”，你们想要的 Java 学习资料来啦！

不管你是学生，还是已经步入职场的同行，希望你们都要珍惜眼前的学习机会，奋斗没有终点，知识永不过时。

扫描下方二维码即可领取

乐字节晓啡

乐字节官方交流群

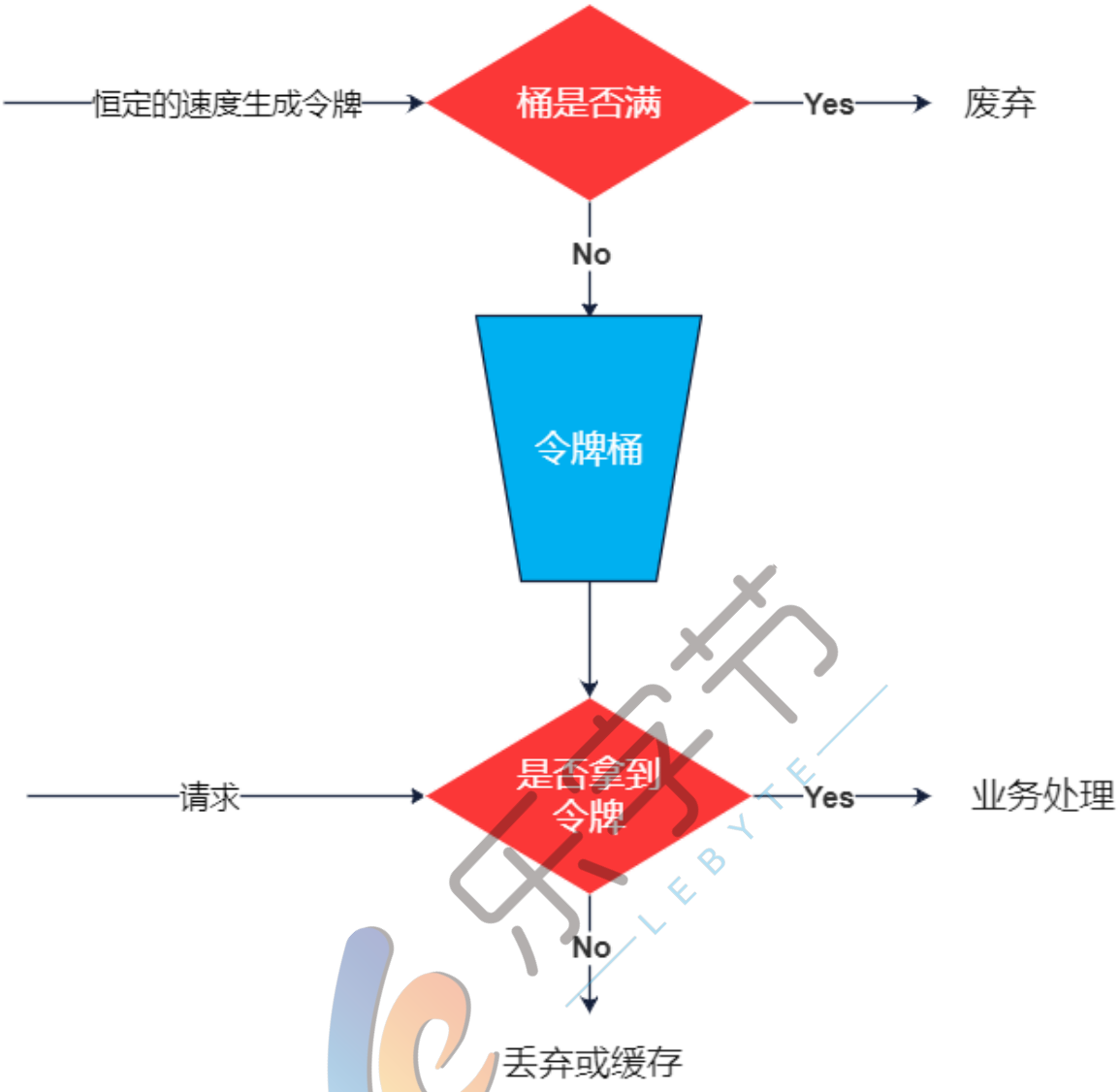
令牌桶算法

令牌桶算法是对漏桶算法的一种改进，漏桶算法能够限制请求调用的速率，而令牌桶算法能够在限制调用的平均速率的同时还允许一定程度的突发调用。在令牌桶算法中，存在一个桶，用来存放固定数量的令牌。算法中存在一种机制，以一定的速率往桶中放令牌。每次**请求调用需要先获取令牌**，只有**拿到令牌**，才有机会**继续执行**，否则选择选择等待可用的令牌、或者直接拒绝。放令牌这个动作是持续不断的进行，如果桶中令牌数达到上限，就丢弃令牌。

> 场景大概是这样的：桶中一直有大量的可用令牌，这时进来的请求可以直接拿到令牌执行，比如设置 QPS 为 100/s，那么限流器初始化完成一秒后，桶中就已经有 100 个令牌了，等服务启动完成对外提供服务时，该限流器可以抵挡瞬时的 100 个请求。当桶中没有令牌时，请求会进行等待，最后相当于以一定的速率执行。

Spring Cloud Gateway 内部使用的就是该算法，大概描述如下：

- 所有的请求在处理之前都需要拿到一个可用的令牌才会被处理；
- 根据限流大小，设置按照一定的速率往桶里添加令牌；
- 桶设置最大的放置令牌限制，当桶满时、新添加的令牌就被丢弃或者拒绝；
- 请求到达后首先要获取令牌桶中的令牌，拿着令牌才可以进行其他的业务逻辑，处理完业务逻辑之后，将令牌直接删除；
- 令牌桶有最低限额，当桶中的令牌达到最低限额的时候，请求处理完之后将不会删除令牌，以此保证足够的限流。



漏桶算法主要用途在于保护它人，而令牌桶算法主要目的在于保护自己，将请求压力交由目标服务处理。假设突然进来很多请求，只要拿到令牌这些请求会瞬时被处理调用目标服务。

添加依赖

Zuul 的限流保护需要额外依赖 spring-cloud-zuul-ratelimit 组件，限流数据采用 Redis 存储所以还要添加 Redis 组件。

RateLimit 官网文档：https://github.com/marcosbarbero/spring-cloud-zuul-ratelimit

```
<!-- spring cloud zuul ratelimit 依赖 -->
<dependency>
    <groupId>com.marcosbarbero.cloud</groupId>
    <artifactId>spring-cloud-zuul-ratelimit</artifactId>
    <version>2.3.0.RELEASE</version>
```

```
</dependency>
<!-- spring boot data redis 依赖 -->
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-redis</artifactId>
</dependency>
<!-- commons-pool2 对象池依赖 -->
<dependency>
    <groupId>org.apache.commons</groupId>
    <artifactId>commons-pool2</artifactId>
</dependency>
```