Hystrix

1: 概念

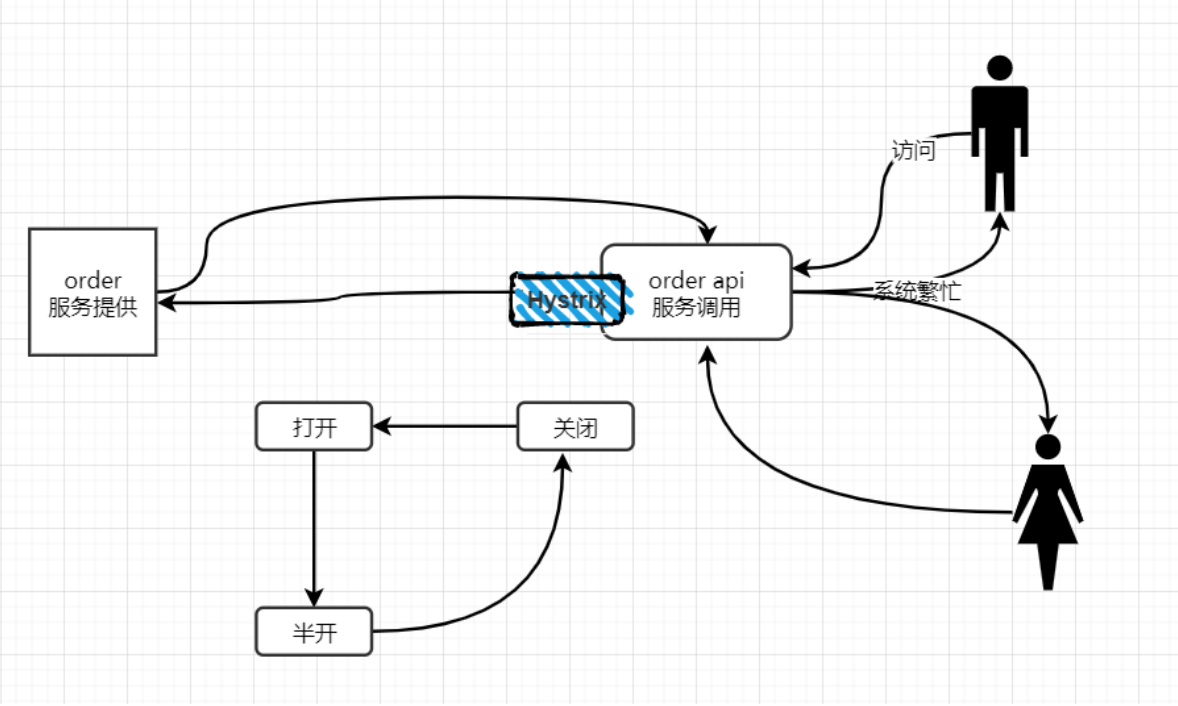
> Hystrix：英 [hɪst'rɪks] 美 [hɪst'rɪks] ，翻译过来是“豪猪”的意思。 在分布式环境中，不可避免地会出现某些依赖的服务发生故障的情况。Hystrix是这样的一个库，它通过添加容许时延和容错逻辑来帮助你控制这些分布式服务之间的交互。Hystrix通过隔离服务之间的访问点，阻止跨服务的级联故障，并提供了退路选项，所有这些都可以提高系统的整体弹性。

2: 服务降级

> 当我们的服务调用者,去调用服务的时候;一定的时间内,如果被调用的没有反应;自己直接返回了;

3: 服务熔断

> 当我们出发了熔断以后; 服务调用者不会去调用服务提供者;
>
> 在一个周期里面所有的请求都不会再去调用了;
>
> 周期以后会重新发送一个请求服务提供者;如果提供者正常响应了,会关闭熔断器;没有,熔断器继续开启;



4： 使用

4.1 导入jar包

```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-netflix-hystrix</artifactId>
</dependency>
```

4.2 开启hystrix

```
feign:
  hystrix:
    enabled: true

hystrix: 配置我们熔断器请求的毫秒数
  command:
    default:
      execution:
        isolation:
          thread:
            timeoutInMilliseconds: 2000
```

4.3 项目启动的时候开启

```
@SpringBootApplication
//@EnableDiscoveryClient  //代表我当前这个微服务是一个服务的消费者 调用别的服务
@EnableFeignClients
@EnableHystrix //启用熔断器
public class OrderApiApplication {

    public static void main(String[] args) {
        SpringApplication.run(OrderApiApplication.class, args);
    }

}
```

4.4 设置我们发生服务降级的回调处理

```
@FeignClient(name = "ldx-order",fallback = OrderApiClientFallBack.class)
public interface OrderFeignClient {

    @PostMapping("/order/add")
    public OrderInfo add(OrderInfo orderInfo);

}
```

4.5 需要一个回调类

```
package com.ldx.fallback;

import com.ldx.info.OrderInfo;
import com.ldx.webfeign.OrderFeignClient;
import org.springframework.stereotype.Component;

@Component
public class OrderApiClientFallBack implements OrderFeignClient {

    //如果这个类；需要执行方法；执行接口里面的同名方法
    @Override
    public OrderInfo add(OrderInfo orderInfo) {
        System.out.println("发生了服务降级");
        return null;
    }
}
```