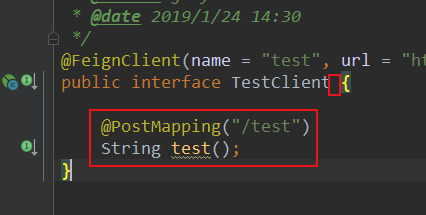
# 熔断器Hystrix

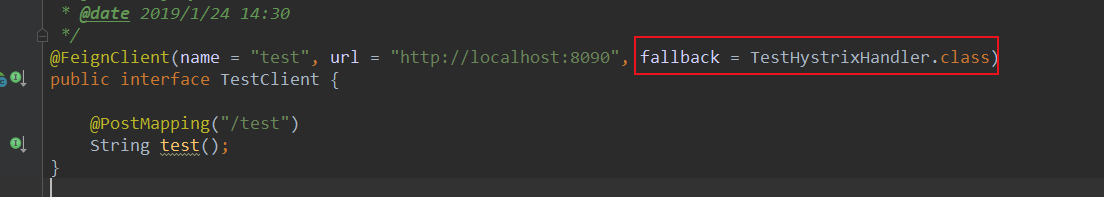
在分布式架构中，当某个服务单元发生故障（类似用电器发生短路）之后，通过断路器的故障监控（类似熔断保险丝），向调用方返回一个错误响应，而不是长时间的等待。这样就不会使得线程因调用故障服务被长时间占用不释放，避免了故障在分布式系统中的蔓延。

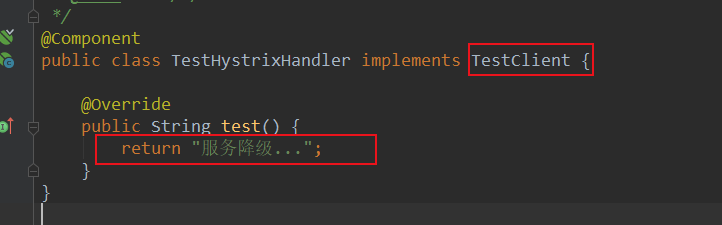
**方式一：**

1. Client接口不再继承service打的jar包中的接口 需要重新接入每个接口的方法 加上mapping路径

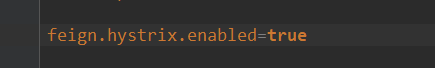


1. @FeignClient加上 fallback属性 值为当前Client接口的实现类 实现类里面的方法内容为该Client每个接口的熔断处理逻辑



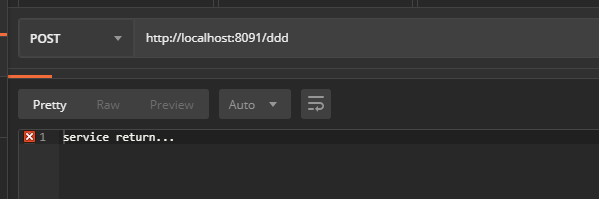


C) application.properties配置熔断处理开启

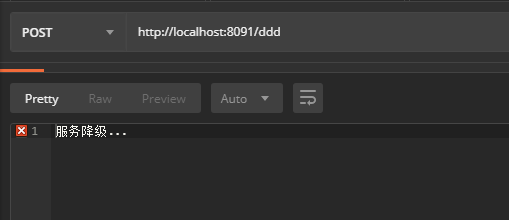


测试：

启动api以及service工程，正常调用

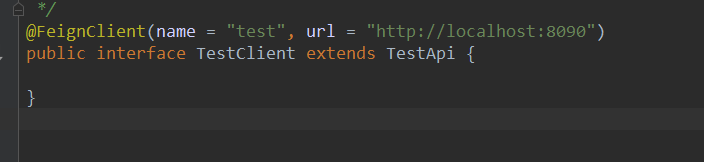


关闭service工程，熔断处理



**方式二：**

1. Client保持不变

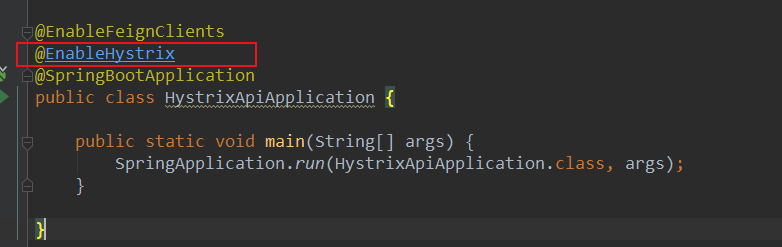


1. Controller类加上@HystrixCommand(fallback=”方法名”)

该方法需要在当前@HystrixCommand注解相同类里，同时需要和@HystrixCommand注解修饰的方法有相同的返回值和入参

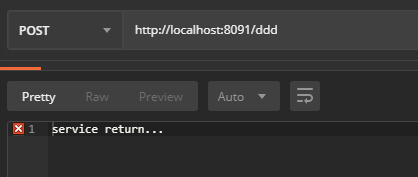


C）springboot启动类添加熔断开启注解

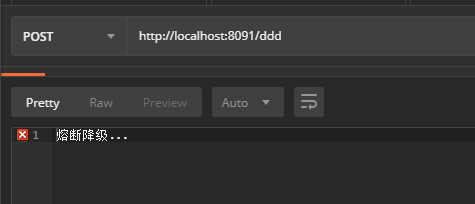


测试：

启动api工程和service工程，正常调用



关闭service工程，熔断处理



第一种方式是针对service工程的接口做熔断处理

第二种方式是针对api工程的接口做熔断处理，如果需要对service接口做熔断处理，需要在service工程中加熔断相关逻辑 @EnableHystrix、 @HystrixCommand(fallbackMethod = "defaultMethod")等

第二种方式如果controller类中有多个接口需要熔断 可以在类上加@DefaultProperties注解

该注解包含以下属性：

**1 groupKey**：一组 Hystrix 命令的集合， 用来统计、报告，默认取类名，可不配置。  
**2 threadPoolKey**：用来标识一个线程池，如果没设置的话会取 groupKey，很多情况下都是同一个类内的方法在共用同一个线程池，如果两个共用同一线程池的方法上配置了同样的属性，在第一个方法被执行后线程池的属性就固定了，所以属性会以第一个被执行的方法上的配置为准。

**3 commandProperties:**

由 @HystrixProperty 注解数组构成的，HystrixProperty 由 name 和 value 两个属性，数据类型都是字符串。

 **execution.isolation.strategy**： 配置请求隔离的方式，有 threadPool（线程池，默认）和 semaphore（信号量）两种，信号量方式高效但配置不灵活，我们一般采用 Java 里常用的线程池方式。

 **execution.timeout.enabled**：是否给方法执行设置超时，默认为 true。

 **execution.isolation.thread.timeoutInMilliseconds**：方法执行超时时间，默认值是 1000，即 1秒，此值根据业务场景配置。

 **execution.isolation.thread.interruptOnTimeout**： **execution.isolation.thread.interruptOnCancel**：是否在方法执行超时/被取消时中断方法。需要注意在 JVM 中我们无法强制中断一个线程，如果 Hystrix 方法里没有处理中断信号的逻辑，那么中断会被忽略。

 **execution.isolation.semaphore.maxConcurrentRequests**：默认值是 10，此配置项要在 execution.isolation.strategy 配置为 semaphore 时才会生效，它指定了一个 Hystrix 方法使用信号量隔离时的最大并发数，超过此并发数的请求会被拒绝。信号量隔离的配置就这么一个，也是前文说信号量隔离配置不灵活的原因。

**4 threadPoolProperties**:

**** coreSize：核心线程池的大小，默认值是 10，一般根据 QPS \* 99% cost + redundancy count 计算得出。

 allowMaximumSizeToDivergeFromCoreSize：是否允许线程池扩展到最大线程池数量，默认为 false;

 maximumSize：线程池中线程的最大数量，默认值是 10，此配置项单独配置时并不会生效，需要启用 allowMaximumSizeToDivergeFromCoreSize 项。

 maxQueueSize：作业队列的最大值，默认值为 -1，设置为此值时，队列会使用 SynchronousQueue，此时其 size 为0，Hystrix 不会向队列内存放作业。如果此值设置为一个正的 int 型，队列会使用一个固定 size 的 LinkedBlockingQueue，此时在核心线程池内的线程都在忙碌时，会将作业暂时存放在此队列内，但超出此队列的请求依然会被拒绝。

 queueSizeRejectionThreshold：由于 maxQueueSize 值在线程池被创建后就固定了大小，如果需要动态修改队列长度的话可以设置此值，即使队列未满，队列内作业达到此值时同样会拒绝请求。此值默认是 5，所以有时候只设置了 maxQueueSize 也不会起作用。

 keepAliveTimeMinutes：由上面的 maximumSize，我们知道，线程池内核心线程数目都在忙碌，再有新的请求到达时，线程池容量可以被扩充为到最大数量，等到线程池空闲后，多于核心数量的线程还会被回收，此值指定了线程被回收前的存活时间，默认为 2，即两分钟。

**5 ignoreExceptions**：默认 Hystrix 在执行方法时捕获到异常时执行回退，并统计失败率以修改熔断器的状态，而被忽略的异常则会直接抛到外层，不会执行回退方法，也不会影响熔断器的状态。  
**6 raiseHystrixExceptions**：当配置项包括 HystrixRuntimeException 时，所有的未被忽略的异常都会被包装成 HystrixRuntimeException，配置其他种类的异常好像并没有什么影响。

**7 defaultFallback**：默认回退方法，当配置 fallbackMethod 项时此项没有意义，另外，默认回退方法不能有参数，返回值要与 Hystrix方法的返回值相同。

测试：  
