服务熔断

极客时间

扫码试看/订阅

《玩转 Spring 全家桶》

使用 Hystrix 实现服务熔断

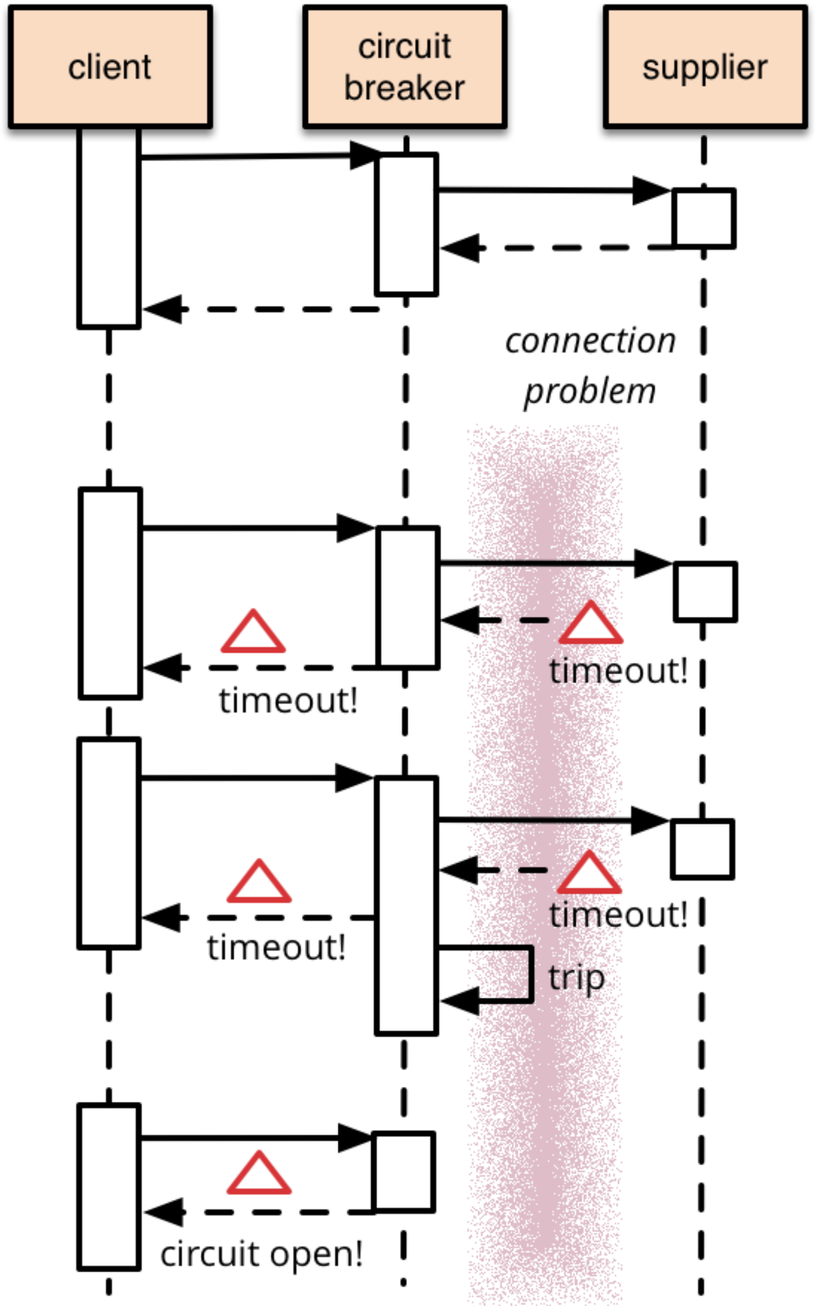
断路器模式

断路器

- Circuit Breaker pattern - Release It, Michael Nygard
- CircuitBreaker, Martin Fowler
  - https://martinfowler.com/bliki/CircuitBreaker.html

核心思想

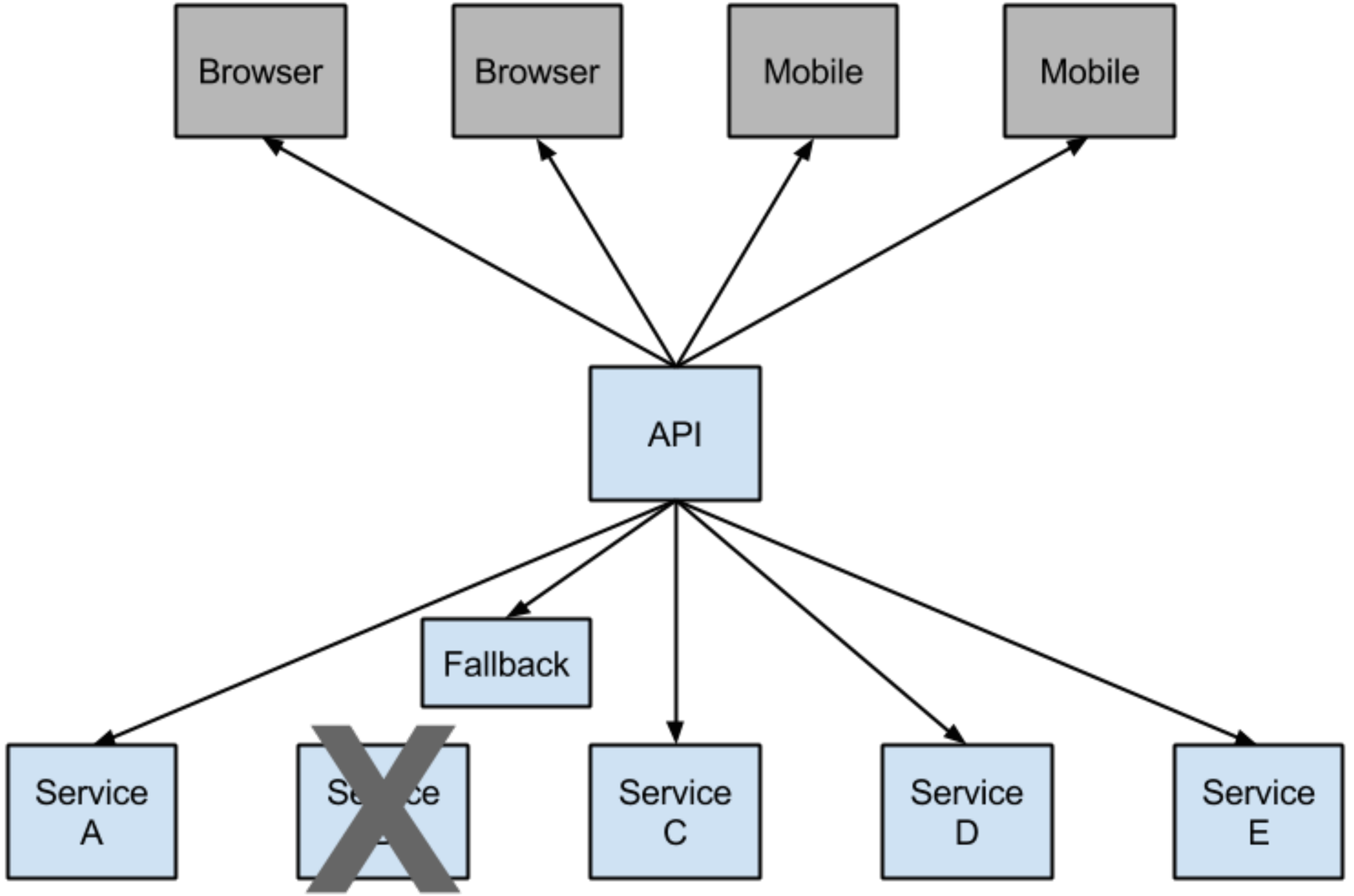
- 在断路器对象中封装受保护的方法调用
- 该对象监控调用和断路情况
- 调用失败触发阈值后，后续调用直接由断路器返回错误，不再执行实际调用


client
circuit breaker
supplier
connection problem
timeout!
timeout!
timeout!
timeout!
trip
circuit open!

“Talk is cheap, show me the code.”

Chapter 13 / circuit-break-demo

Netflix Hystrix



Netflix Hystrix

- 实现了断路器模式
- @HystrixCommand
  - fallbackMethod / commandProperties
    - @HystrixProperty(name="execution.isolation.strategy", value="SEMAPHORE")
  - https://github.com/Netflix/Hystrix/wiki/Configuration

Netflix Hystrix

Spring Cloud 支持

- spring-cloud-starter-netflix-hystrix
- `@EnableCircuitBreaker`

Feign 支持

- `feign.hystrix.enabled=true`
- `@FeignClient`
  - `fallback / fallbackFactory`

“Talk is cheap, show me the code.”

Chapter 13 / hystrix-customer-service

如何观察服务熔断

如何了解熔断的情况

打日志

- 在发生熔断时打印特定该日志

看监控

- 主动向监控系统埋点，上报熔断情况
- 提供与熔断相关的 Endpoint，让第三方系统来拉取信息

Hystrix Dashboard

Spring Cloud 为我们提供了

- Hystrix Metrics Stream
  - spring-boot-starter-actuator
    - /actuator/hystrix.stream
- Hystrix Dashboard
  - spring-cloud-starter-netflix-hystrix-dashboard
    - `@EnableHystrixDashboard`
    - /hystrix

“Talk is cheap, show me the code.”

Chapter 13 / hystrix-stream-customer-service hystrix-dashboard-demo

聚合集群熔断信息

Spring Cloud 为我们提供了

- Netflix Turbine
  - spring-cloud-starter-netflix-turbines
    - `@EnableTurbine`
    - /turbine.stream?cluster=集群名

“Talk is cheap, show me the code.”

Chapter 13 / turbine-demo

使用 Resilience4j 实现服务熔断

Hystrix 以外的选择

Hystrix

- Netflix 停止维护，给了官方推荐

Resilience4j

- https://github.com/resilience4j/resilience4j
- 一款受 Hystrix 启发的轻量级且易于使用的容错库
- 针对 Java 8 与函数式编程设计

核心组件

组件名称	功能
resilience4j-circuitbreaker	Circuit breaking
resilience4j-ratelimiter	频率控制
resilience4j-bulkhead	依赖隔离&负载保护
resilience4j-retry	自动重试
resilience4j-cache	应答缓存
resilience4j-timelimiter	超时控制

附加组件

组件名称	功能
resilience4j-reactor	Spring Reactor 支持
resilience4j-micrometer	Micrometer Metrics 输出
resilience4j-prometheus	Prometheus Metrics 输出
resilience4j-spring-boot2	Spring Boot 2 Starter
resilience4j-feign	Feign 适配器

断路器

实现

- 基于 ConcurrentHashMap 的内存断路器
- CircuitBreakerRegistry
- CircuitBreakerConfig

依赖

- resilience4j-spring-boot2
  - resilience4j-circuitbreaker
  - resilience4j-micrometer
  - ……

断路器

注解方式

- @CircuitBreaker(name = "名称")

配置

- CircuitBreakerProperties
  - resilience4j.circuitbreaker.backends.名称
    - failure-rate-threshold
    - wait-duration-in-open-state

“Talk is cheap, show me the code.”

Chapter 13 / resilience4j-circuitbreaker-demo

使用 Resilience4j 实现服务限流

Bulkhead

目的

- 防止下游依赖被并发请求冲击
- 防止发生连环故障

用法

- `BulkheadRegistry / BulkheadConfig`
- `@Bulkhead(name = "名称")`

Bulkhead

配置

- BulkheadProperties
  - resilience4j.bulkhead.backends.名称
    - max-concurrent-call
    - max-wait-time

“Talk is cheap, show me the code.”

Chapter 13 / bulkhead-customer-service

RateLimiter

目的

- 限制特定时间段内的执行次数

用法

- `RateLimiterRegistry / RateLimiterConfig`
- `@RateLimiter(name = "名称")`

RateLimiter

配置

- RateLimiterProperties
  - resilience4j.ratelimiter.limiters.名称
    - limit-for-period
    - limit-refresh-period-in-millis
    - timeout-in-millis

“Talk is cheap, show me the code.”

Chapter 13 / ratelimiter-waiter-service

SpringBucks 实战项目进度小结

本章小结

几种模式

- 断路器 / 隔舱 / 速率限制器

两个工具

- Netflix Hystrix / Resilience4j
- 建议学习，Google Guava

观察与监控

- Hystrix Dashboard / Micrometer

SpringBucks 进度小结

waiter-service

- 使用 Resilience4j 的 `RateLimiter` 进行防护

customer-service

- 使用 Hystrix 进行熔断
- 使用 Resilience4j 进行熔断和并发控制

极客时间

扫码试看/订阅

《玩转 Spring 全家桶》