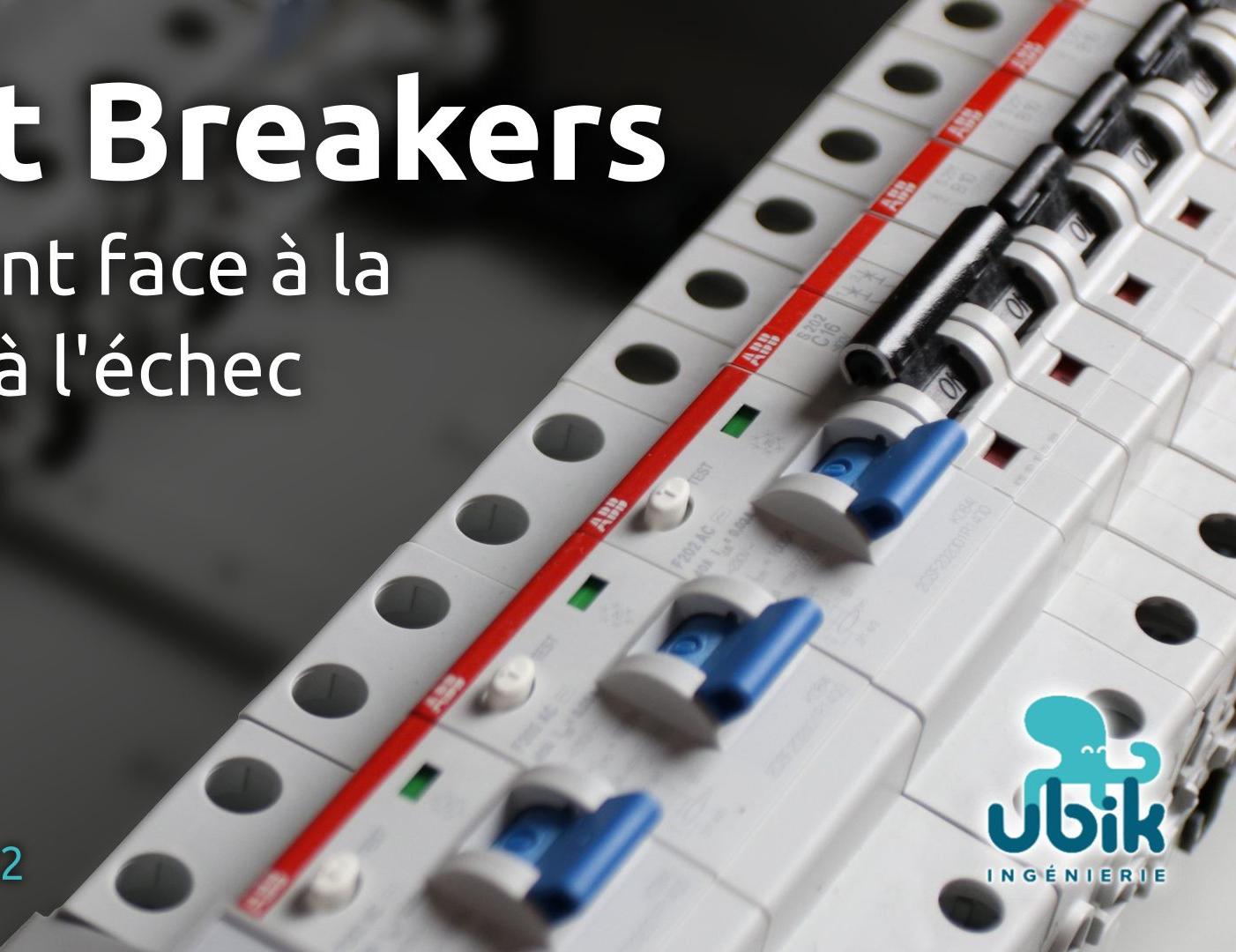


Circuit Breakers

Être résilient face à la latence et à l'échec

Romain V.
mardi 12 juillet 2022



Qui suis-je

Romain V.

Chez Ubik Ingénierie depuis septembre 2021

Développeur Java - Angular

Projets pour différents clients



Plan

Design pattern

Resilience4j

Configuration

Exemple d'utilisation

Des questions ?
Prenez-note :

Design pattern



De nouvelles problématiques

THE EVOLUTION OF

SOFTWARE ARCHITECTURE

1990's

SPAGHETTI-ORIENTED
ARCHITECTURE
(aka Copy & Paste)



2000's

LASAGNA-ORIENTED
ARCHITECTURE
(aka Layered Monolith)



2010's

RAVIOLI-ORIENTED
ARCHITECTURE
(aka Microservices)

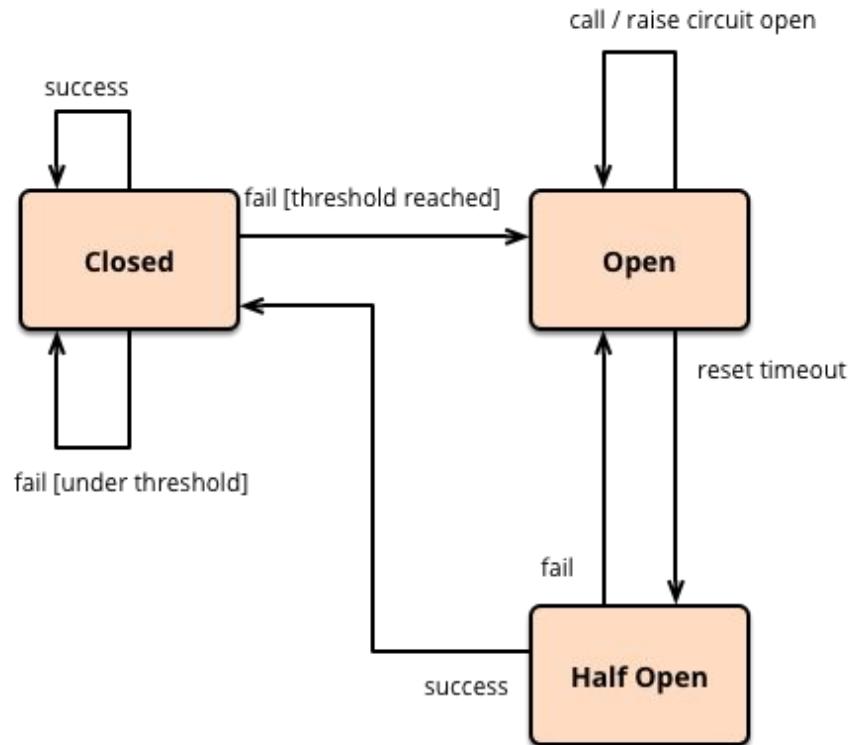


WHAT'S NEXT?

PROBABLY PIZZA-ORIENTED ARCHITECTURE



Design pattern



Resilience4j



Utilisation avec spring



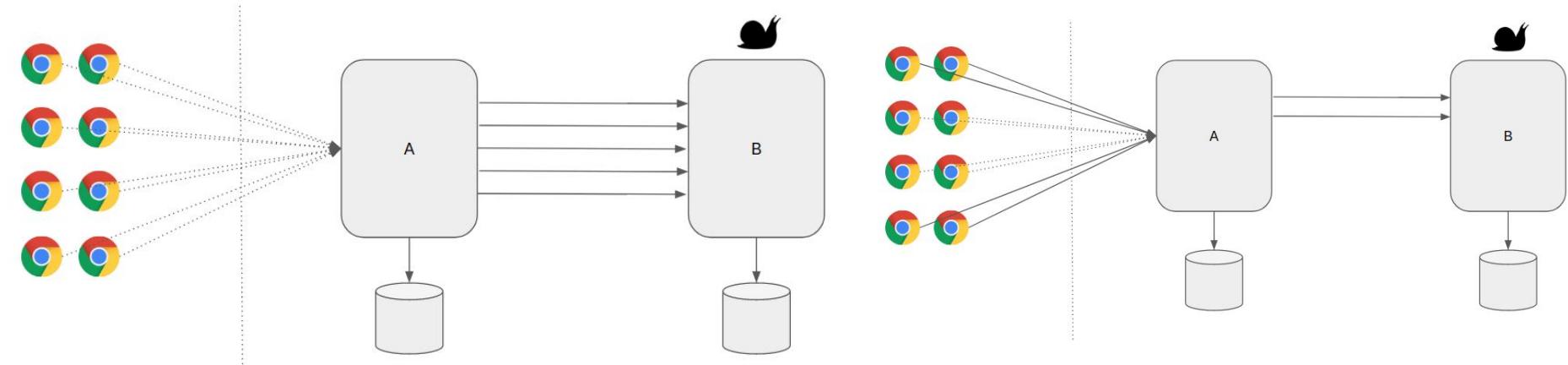
- io.github.resilience4j:resilience4j-spring-boot2
- org.springframework.boot:spring-boot-starter-actuator
- org.springframework.boot:spring-boot-starter-aop

Les dépendances de resilience4j

- ▼ resilience4j-spring-boot2 : 1.7.1 [compile]
 - ▶ vavr : 0.10.2 [compile]
 - ▼ resilience4j-spring : 1.7.1 [compile]
 -
 - ▶ resilience4j-annotations : 1.7.1 [compile]
 - ▶ resilience4j-consumer : 1.7.1 [compile]
 - ▼ resilience4j-framework-common : 1.7.1 [compile]
 -
 - ▶ resilience4j-circuitbreaker : 1.7.1 [compile]
 -
 -
 -
 -
 - ▶ resilience4j-ratelimiter : 1.7.1 [compile]
 - ▶ resilience4j-retry : 1.7.1 [compile]
 - ▶ resilience4j-bulkhead : 1.7.1 [compile]
 - ▶ resilience4j-timelimiter : 1.7.1 [compile]
 -
 -
 -
 - ▶ resilience4j-micrometer : 1.7.1 [runtime]

Bulkhead

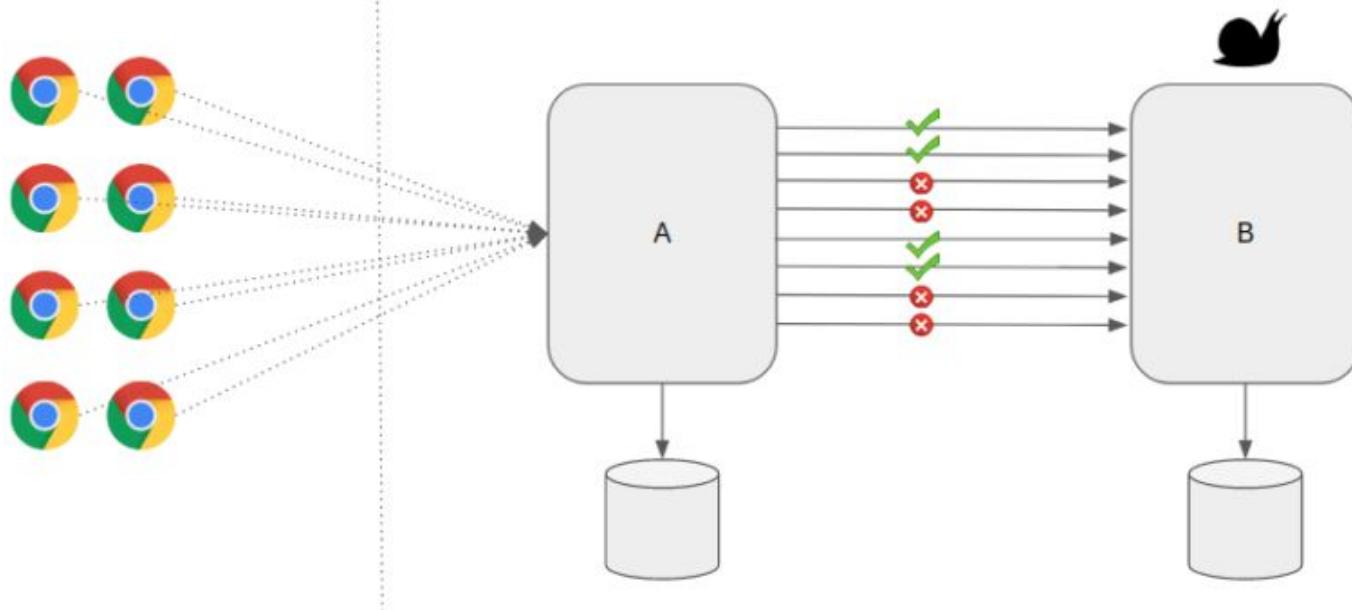
A droite, on a limité à 2 appels concurrents de B (qui est lent) pour éviter de saturer les threads de traitement de B qui est limité en ressources



Annotation resilience4j : @Bulkhead(name=BULKHEAD_NAME)

Ratelimiter

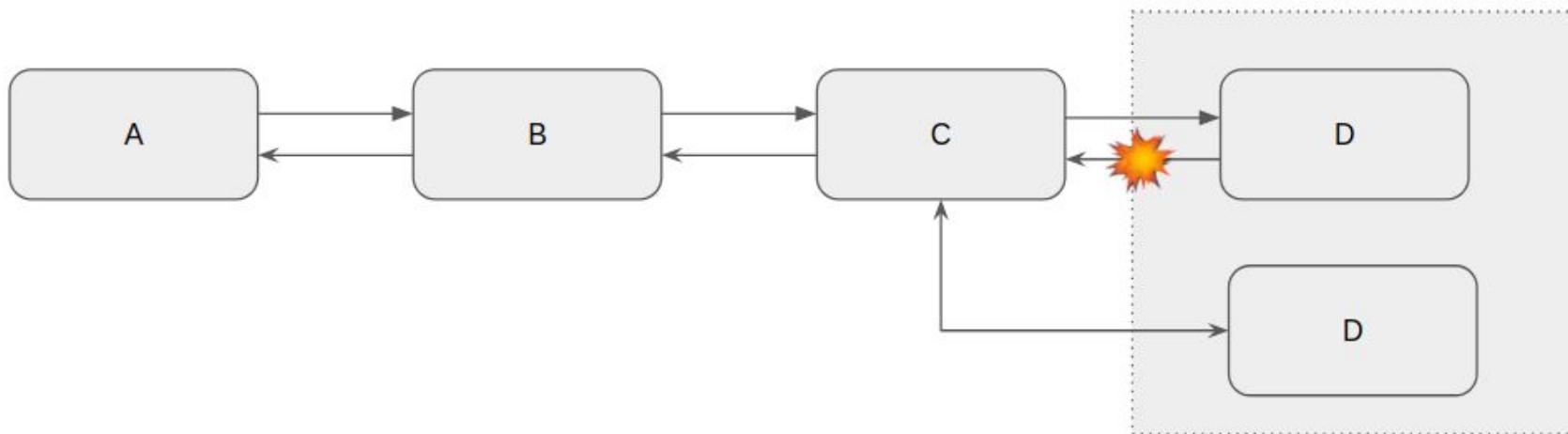
B ne pouvant encaisser plus de X appels sur une durée, on va limiter le nombre d'appels sur une période donnée et rejeter les autres appels plutôt que les traiter et crasher



Annotation resilience4j : @RateLimiter(name=RATELIMITER_NAME)

Retry

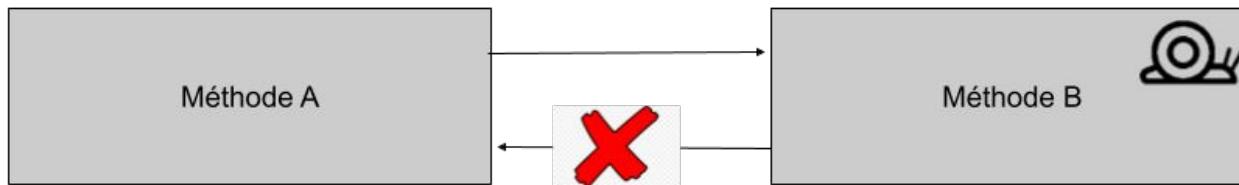
Le premier appel à B a échoué et le second a renvoyé un succès



Annotation resilience4j : @Retry(name=RETRY_NAME)

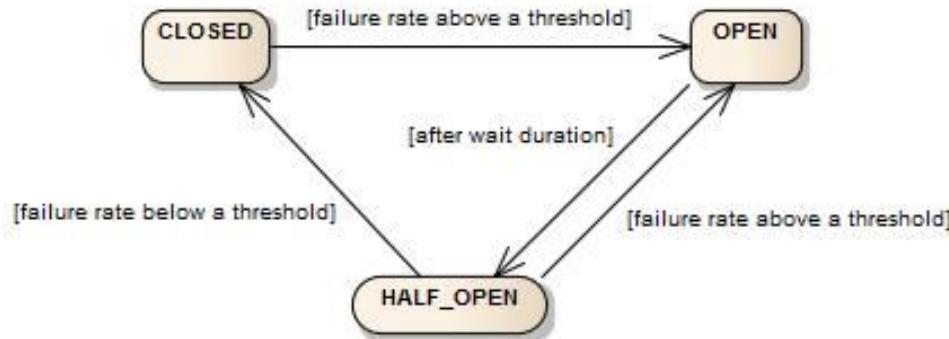
Timelimiter

La méthode B a mis trop de temps à renvoyer un résultat



Annotation resilience4j : @Timelimiter(name=TIMELIMITER_NAME)

Circuit breaker



Gérer la latence et les échecs des webservices utilisés par votre application

Annotation resilience4j : @CircuitBreaker(name=CIRCUIT_BREAKER_NAME)

Configuration

```
129
130
131 .mail{
132   background: url(..../img/mailico.png) no-repeat center;
133   display: inline-block;
134   width: 12px;
135   height: 24px;
136   float: left;
137   margin-left: 2px 7px 0 0;
138 }
139 .phone{
140   background: url(..../img/phoneico.png) no-repeat center;
141   display: inline-block;
142   width: 20px;
143   height: 18px;
144   float: left;
145   margin-left: 3px 7px 0 0;
146 }
```

(root@privata-var-folders-11:/opt/cross/vmfs3/00000000000000000000000000000000) [1/8] 800b21-8e7e-4d01-bb6f-0a41b86ab11 /usr/css/style.css

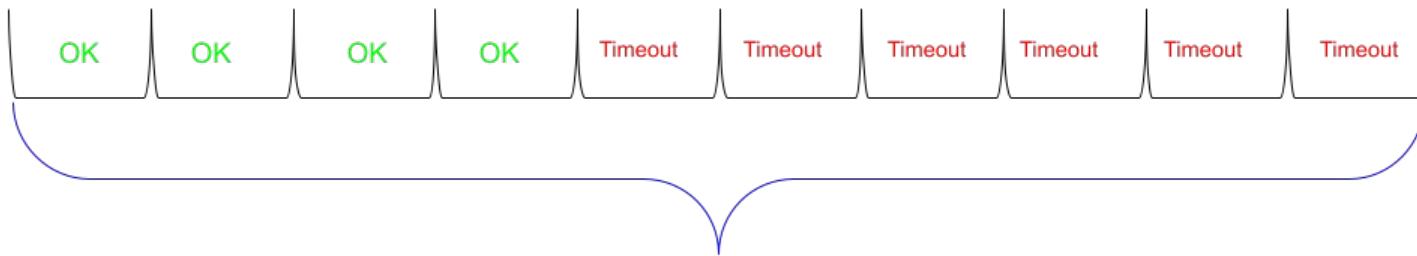
Configuration commune à l'ensemble des circuit breakers

```
resilience4j.circuitbreaker.configs.default.slidingWindowSize=5
resilience4j.circuitbreaker.configs.default.slowCallRateThreshold=60
resilience4j.circuitbreaker.configs.default.slowCallDurationThreshold=1500
resilience4j.circuitbreaker.configs.default.minimumNumberOfCalls=2
resilience4j.circuitbreaker.configs.default.permittedNumberOfCallsInHalfOpenState=1
```

Configuration spécifique à un circuit breaker

```
resilience4j.circuitbreaker.instances.pricebreaker.baseConfig=default
resilience4j.circuitbreaker.instances.pricebreaker.slidingWindowSize=10
resilience4j.circuitbreaker.instances.pricebreaker.slowCallRateThreshold=80
resilience4j.circuitbreaker.instances.pricebreaker.slowCallDurationThreshold=1000
resilience4j.circuitbreaker.instances.pricebreaker.minimumNumberOfCalls=5
resilience4j.circuitbreaker.instances.pricebreaker.permittedNumberOfCallsInHalfOpenState=2
resilience4j.circuitbreaker.instances.pricebreaker.recordExceptions[0]=myapp.service.common.CustomException
resilience4j.circuitbreaker.instances.pricebreaker.recordExceptions[1]=org.springframework.web.client.RestClientException
```

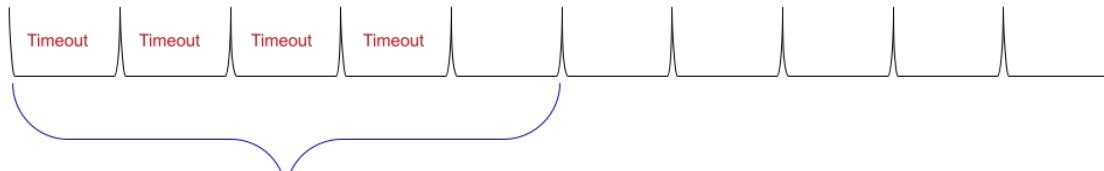
slidingWindowSize



```
resilience4j.circuitbreaker.instances.pricebreaker.slidingWindowSize=10
```

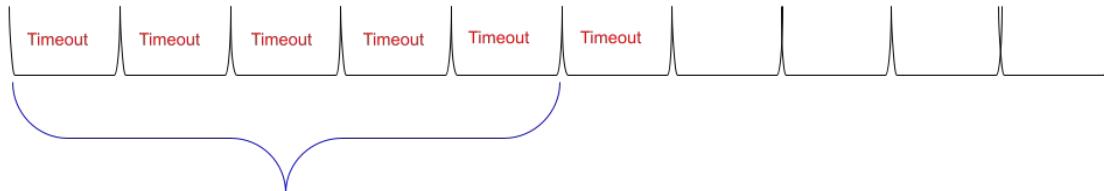
minimumNumberOfCalls

Le circuit breaker n'est pas opérationnel



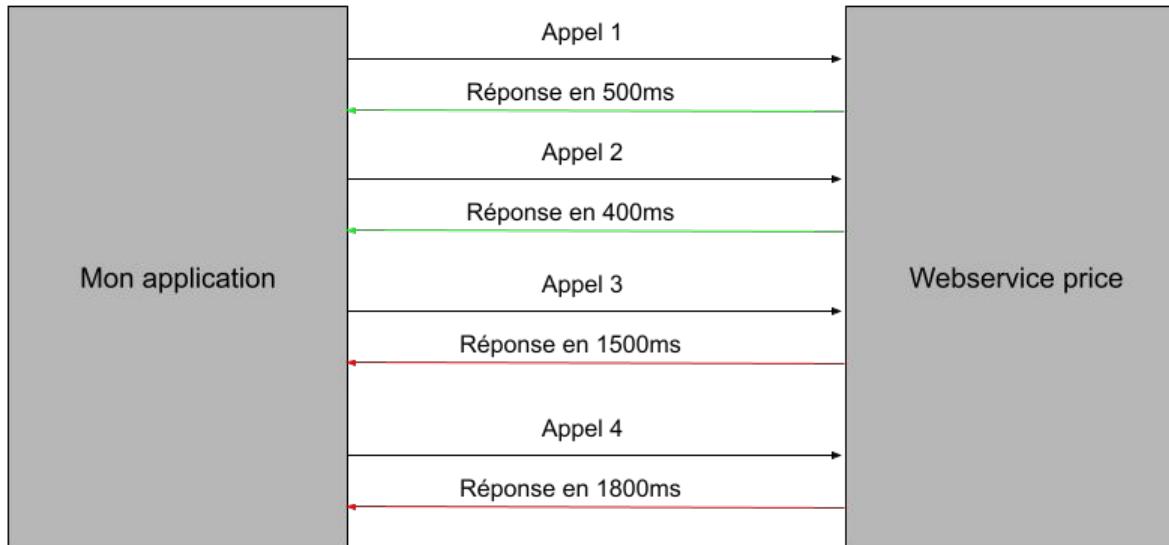
`resilience4j.circuitbreaker.instances.pricebreaker.minimumNumberOfCalls=5`

Le circuit breaker est opérationnel



`resilience4j.circuitbreaker.instances.pricebreaker.minimumNumberOfCalls=5`

slowCallDurationThreshold



```
resilience4j.circuitbreaker.instances.pricebreaker.slowCallDurationThreshold=1000
```

slowCallRateThreshold

```
resilience4j.circuitbreaker.instances.pricebreaker.slowCallRateThreshold=80
```

Le circuit breaker reste dans l'état CLOSED

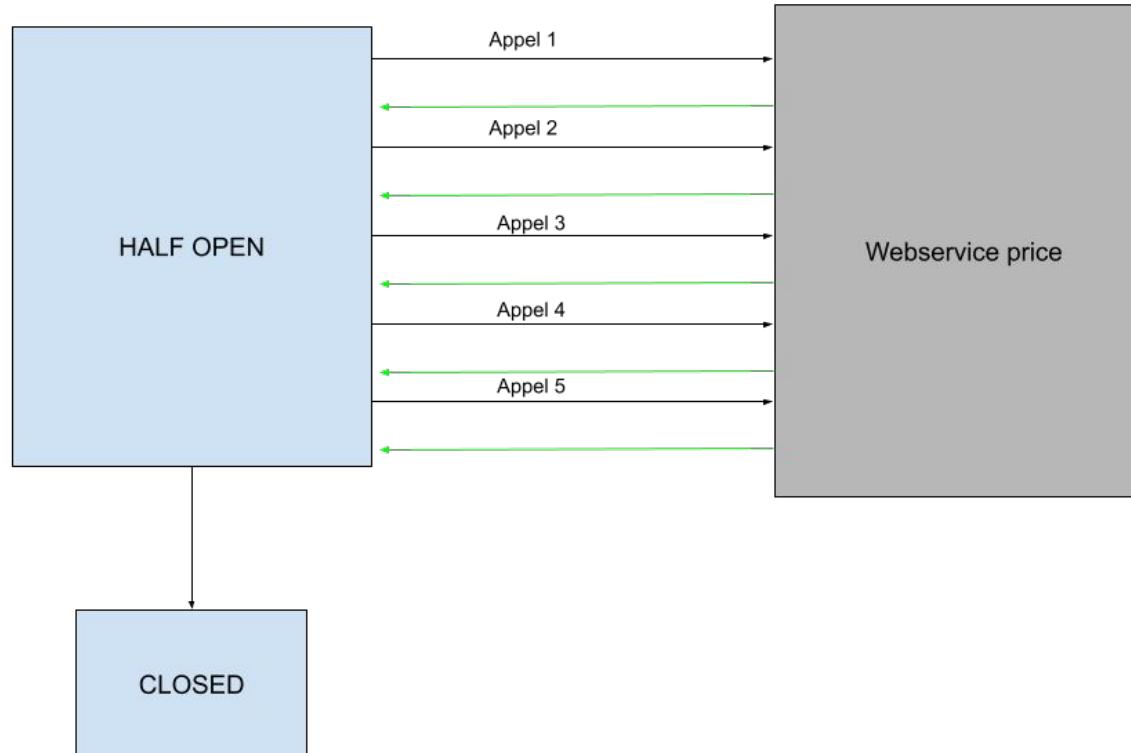


Le circuit breaker passe en OPEN

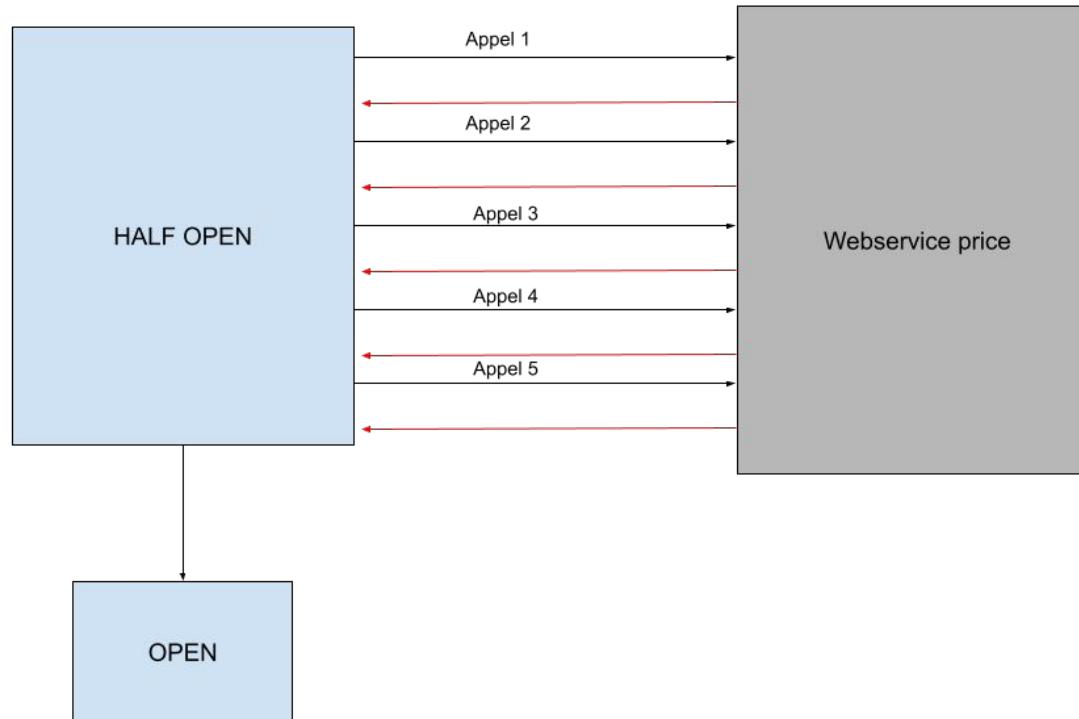


permittedNumberOfCallsInHalfOpenState

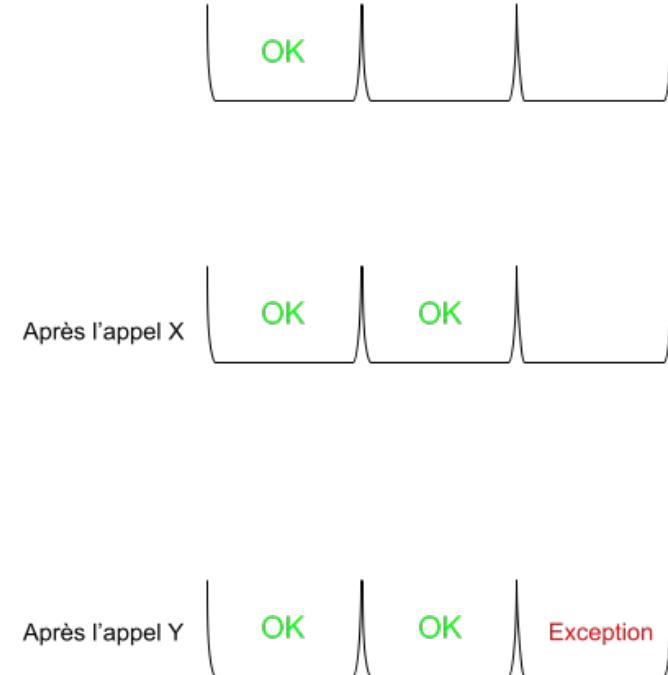
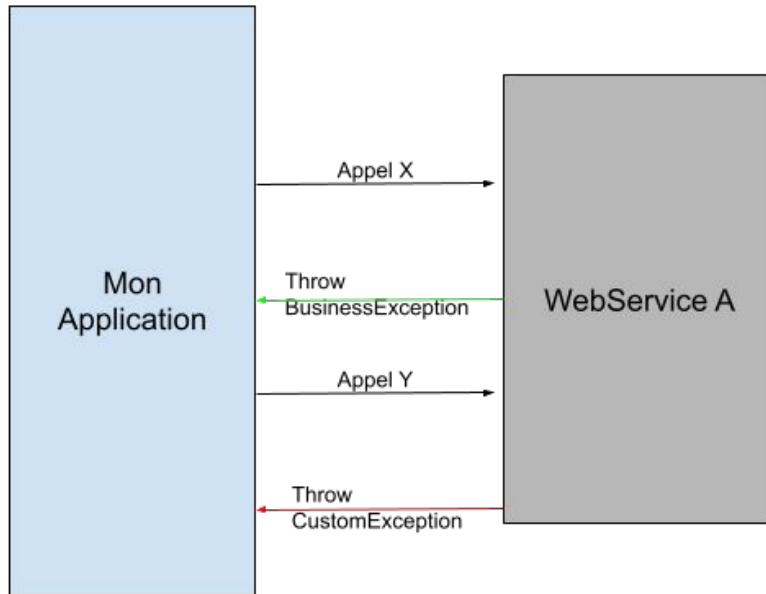
```
resilience4j.circuitbreaker.instances.pricebreaker.minimumNumberOfCalls=5
```



permittedNumberOfCallsInHalfOpenState



recordExceptions[i]



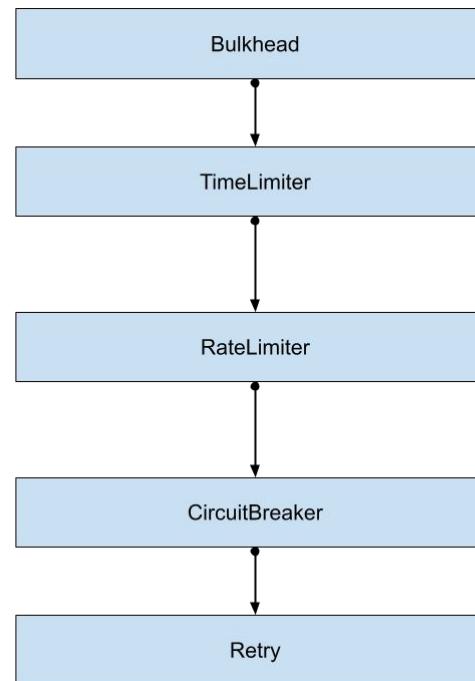
```
resilience4j.circuitbreaker.instances.pricebreaker.recordExceptions[0]=myapp.service.common.CustomException  
resilience4j.circuitbreaker.instances.pricebreaker.recordExceptions[1]=org.springframework.web.client.RestClientException
```

Ensemble des propriétés disponibles

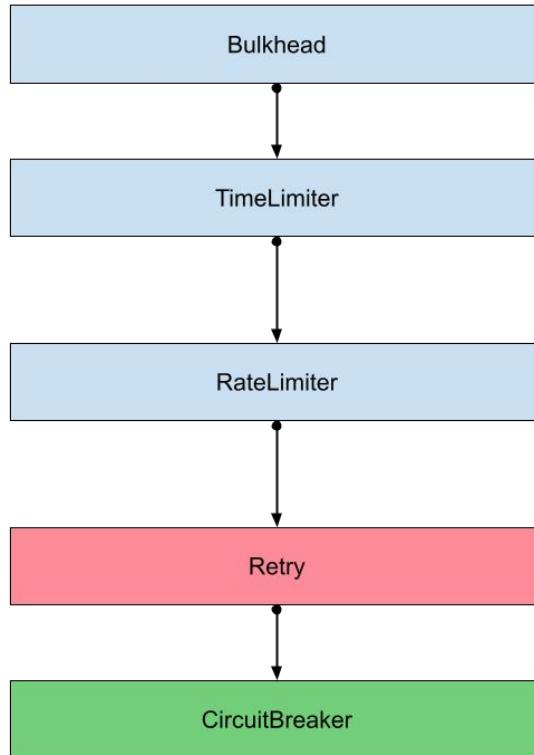
- <https://resilience4j.readme.io/docs/circuitbreaker>

Ordre décorateur

```
@CircuitBreaker(name = "CircuitBreakerServiceA")
@RateLimiter(name = "RateLimiterServiceA")
@Bulkhead(name = "BulkheadServiceA")
@Retry(name = "RetryServiceA")
@TimeLimiter(name = "TimeLimiterServiceA")
public void method(String param1, int param2) {
}
```



Modification ordre décorateur



```
resilience4j.circuitbreaker.circuitBreakerAspectOrder=1  
resilience4j.retry.retryAspectOrder=2
```

Exemple d'utilisation



Exemple circuit breaker (catch exception)

```
@Retry(name = Resilience.RETRY_NAME_FOR_API)
@CircuitBreaker(name = Resilience.CIRCUIT_BREAKER_NAME_FOR_API)
public Response callApi() throws CallNotPermittedException {
    // Imagine some serious job here!
}

try {
    service.callApi();
} catch (CallNotPermittedException e) {
    return new FallbackResponse();
}
```

Exemple circuit breaker (fallback method)

```
@CircuitBreaker(name = Resilience.CIRCUIT_BREAKER_NAME_FOR_API,
    fallbackMethod = "fallbackResponse")
public Response callApi(Some signature) {
    // Imagine some serious job here!
}

private void fallbackResponse(Some sameSignature,
    CallNotPermittedException e) {
    return new FallbackResponse();
}
```



Questions