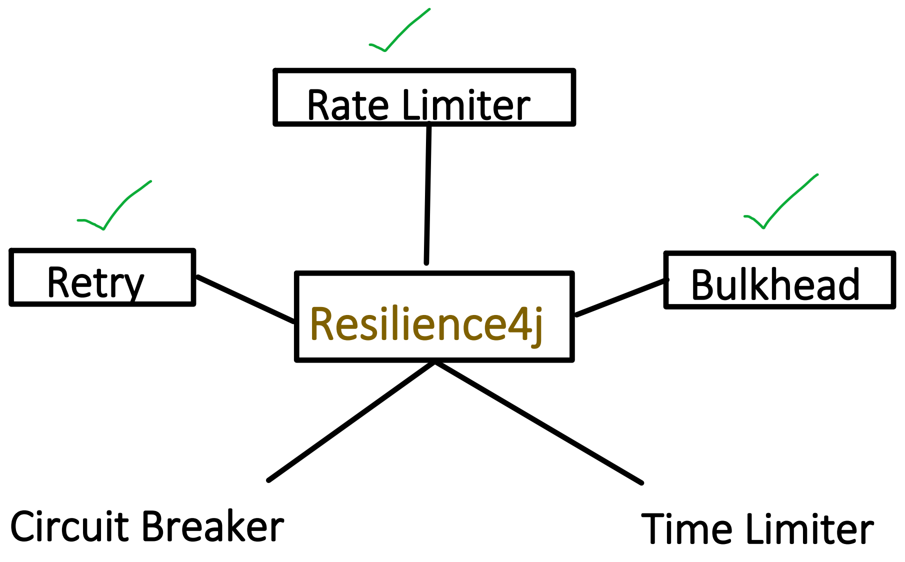
Circuit Breaker: Fault-Tolerant Microservice (Part-4)

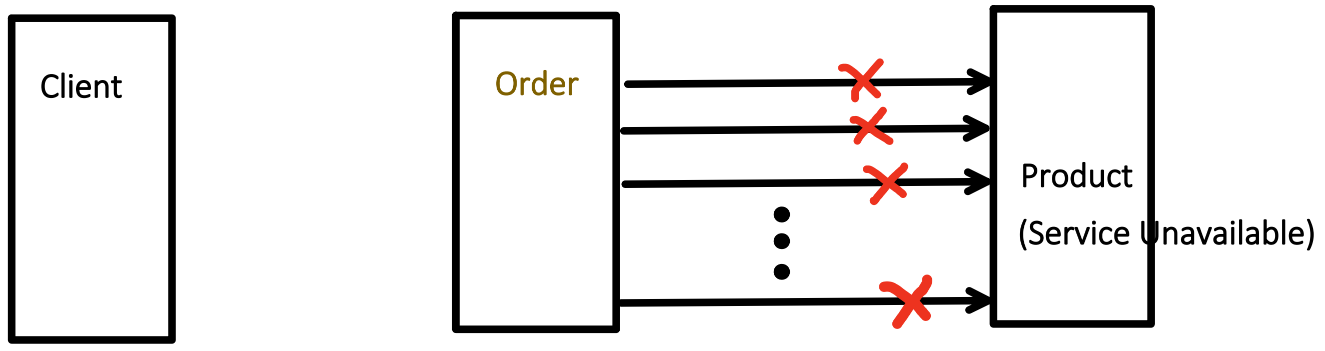
Friday, 1 August 2025 10:07 PM

To build Fault tolerant microservices: Resilience4j provides below mechanisms



Circuit Breaker:

- This pattern prevents an application to make repeated calls to a downstream service that is likely to fail.



Product Service is down, no matter how many times Order service will invoke Product service either by retrying the same call or different call, It will fail.

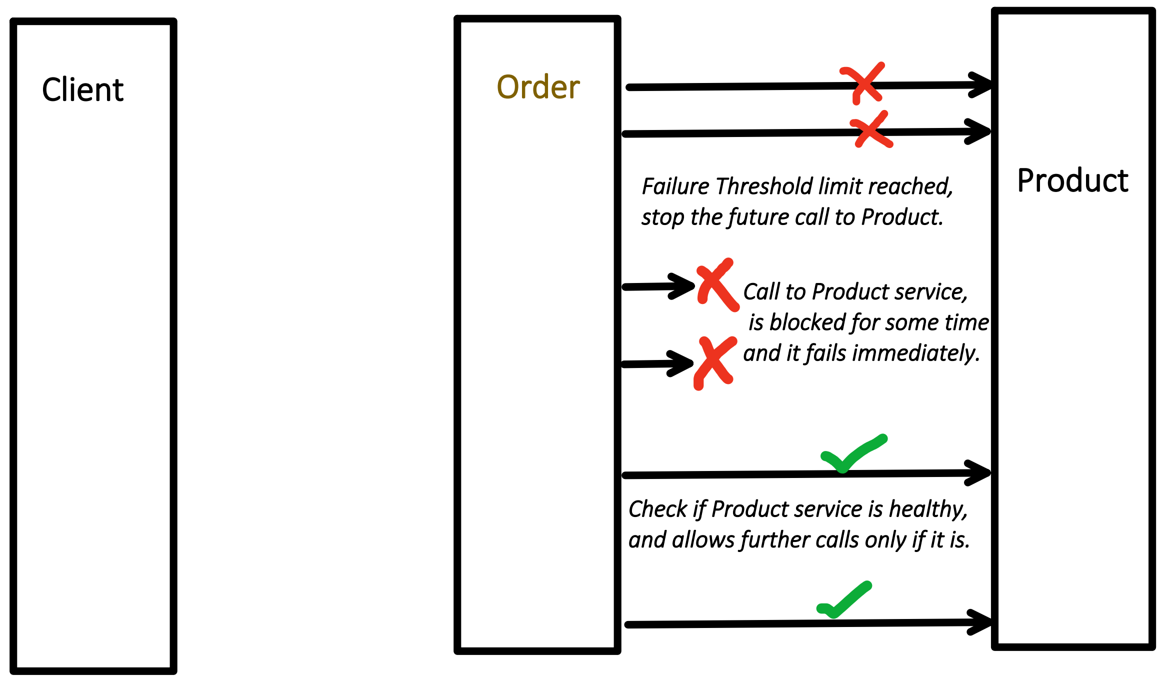
Disadvantages:

- It unnecessarily adds load to Product service and because of that Product service might take longer time to recover.
- Order service is unnecessarily wasting its resource (latency and thread blocking) by making call that is likely going to fail.

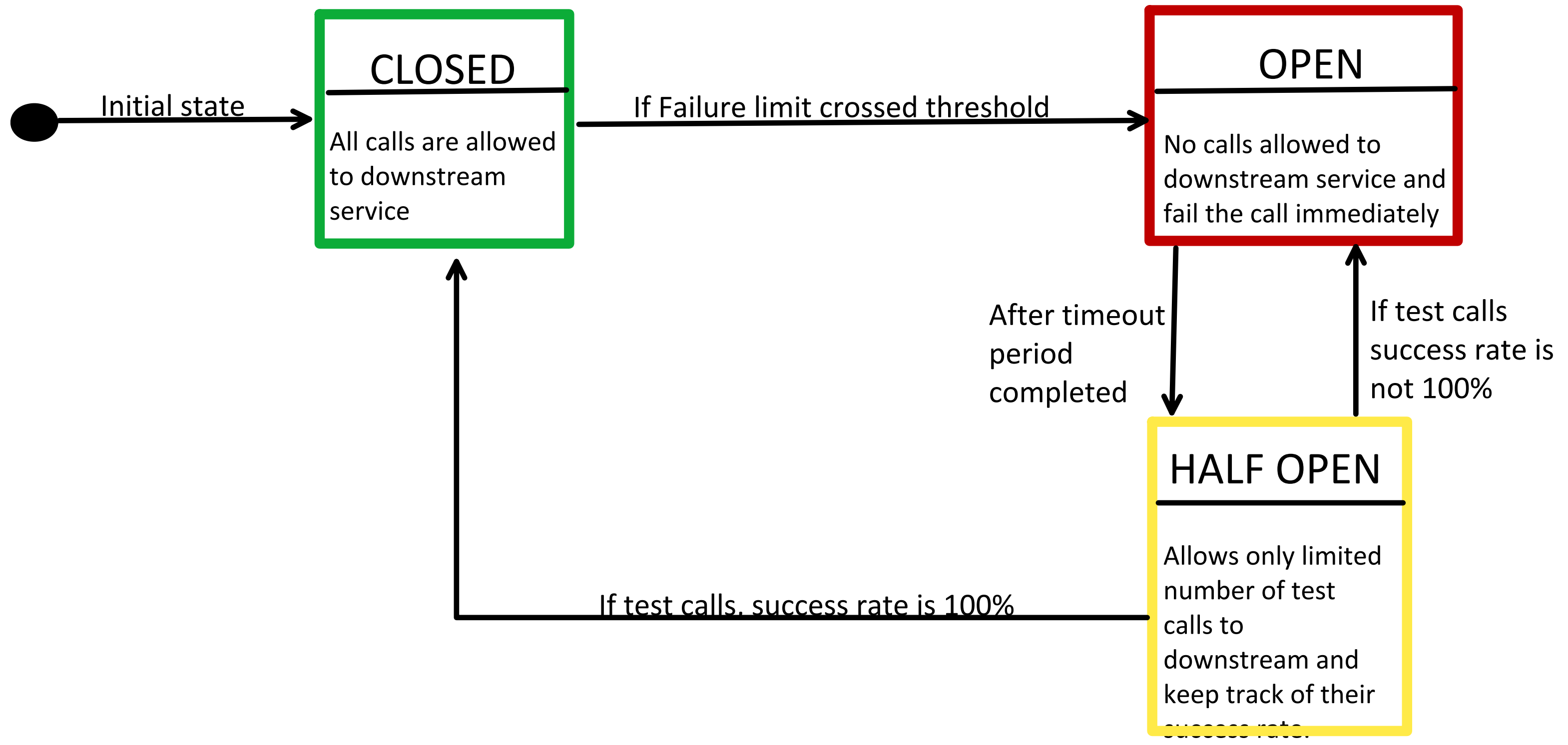
So, what's the solution?
How does Order service know, when to stop the downstream Product service call and when to restart?

Answer is "Circuit Breaker", again, lets revisit its definition.

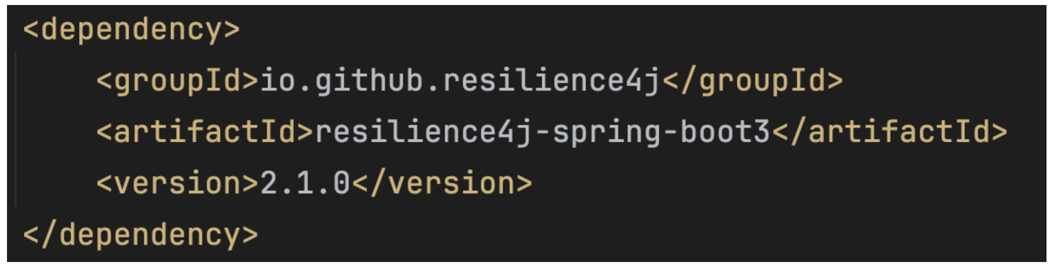
It prevents an application to make repeated calls to a downstream service that is likely to fail.


Client
Order
Product
Failure Threshold limit reached, stop the future call to Product.
Call to Product service, is blocked for some time and it fails immediately.
Check if Product service is healthy, and allows further calls only if it is.

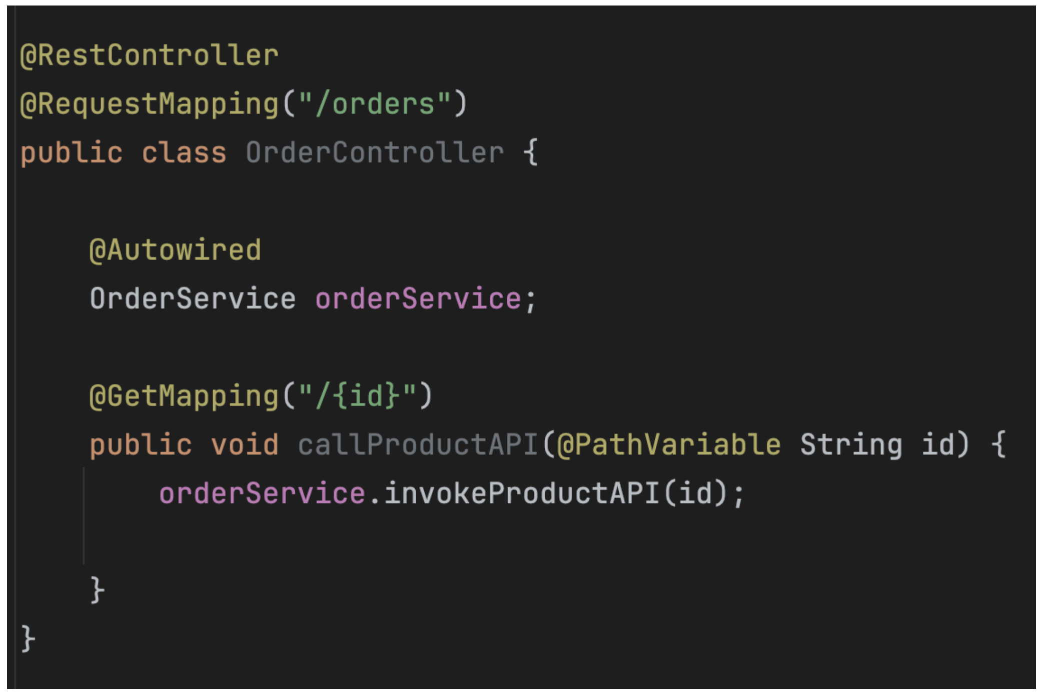
States of Circuit Breaker:


Initial state
CLOSED
All calls are allowed to downstream service
If Failure limit crossed threshold
OPEN
No calls allowed to downstream service and fail the call immediately
After timeout period completed
If test calls success rate is not 100%
HALF OPEN
Allows only limited number of test calls to downstream and keep track of their success rate.
If test calls, success rate is 100%

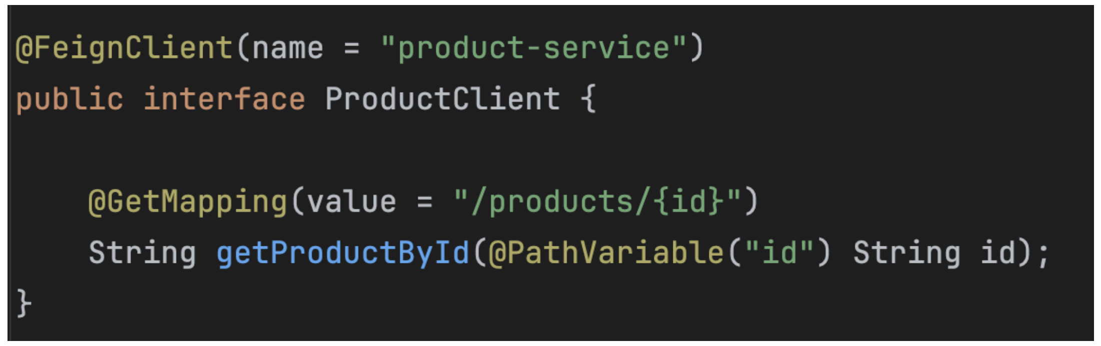
Pom.xml dependency



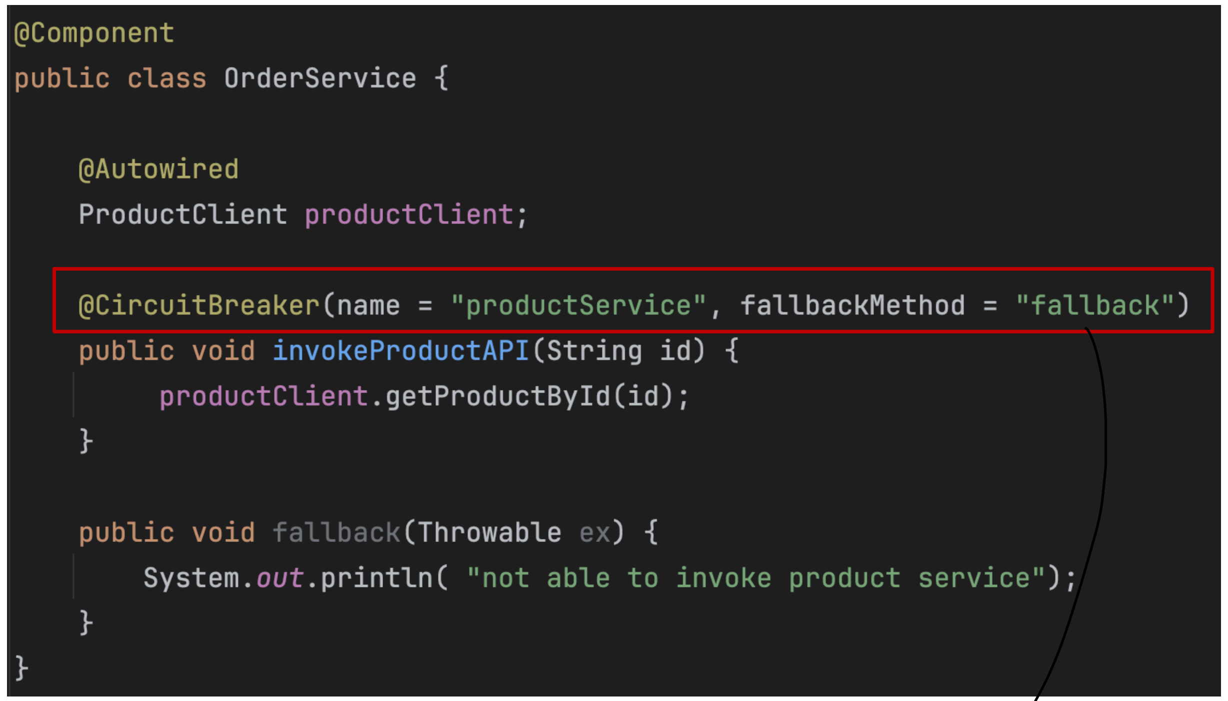
```xml
<dependency>
    <groupId>io.github.resilience4j</groupId>
    <artifactId>resilience4j-spring-boot3</artifactId>
    <version>2.1.0</version>
</dependency>
```



```java
@RestController
@RequestMapping("/orders")
public class OrderController {

    @Autowired
    OrderService orderService;

    @GetMapping("/{id}")
    public void callProductAPI(@PathVariable String id) {
        orderService.invokeProductAPI(id);

    }
}
```



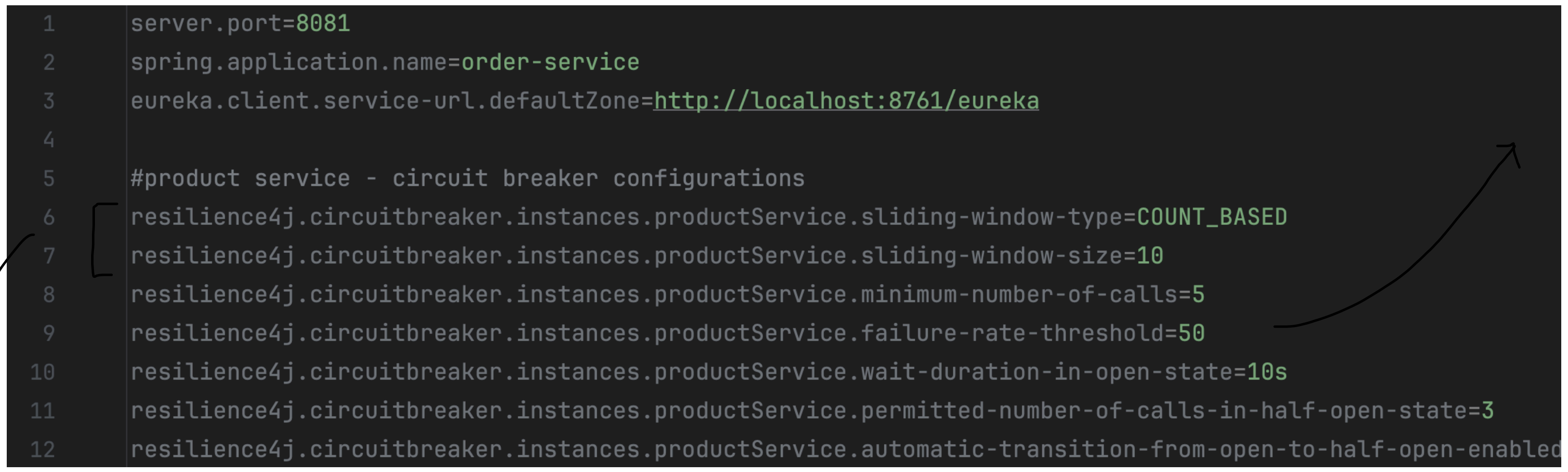
```java
@FeignClient(name = "product-service")
public interface ProductClient {

    @GetMapping(value = "/products/{id}")
    String getProductById(@PathVariable("id") String id);
}
```



```java
@Component
public class OrderService {

    @Autowired
    ProductClient productClient;

    @CircuitBreaker(name = "productService", fallbackMethod = "fallback")
    public void invokeProductAPI(String id) {
        productClient.getProductById(id);
    }

    public void fallback(Throwable ex) {
        System.out.println( "not able to invoke product service");
    }
}
```

Fallback method is invoked for each failure attempt.

application.properties



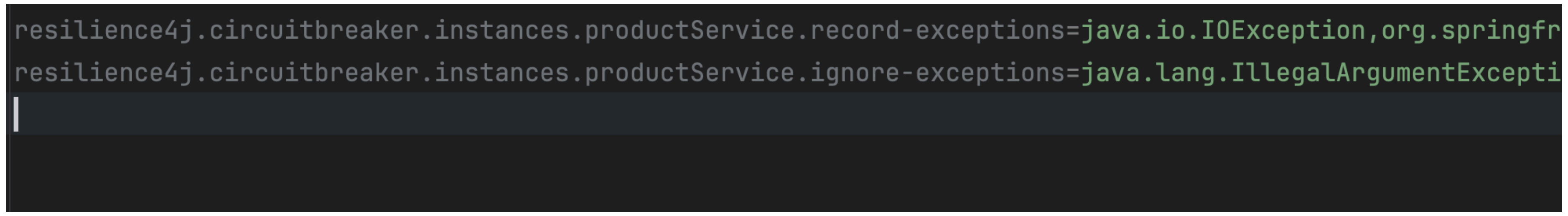
```properties
server.port=8081
spring.application.name=order-service
eureka.client.service-url.defaultZone=http://localhost:8761/eureka

#product service - circuit breaker configurations
resilience4j.circuitbreaker.instances.productService.sliding-window-type=COUNT_BASED
resilience4j.circuitbreaker.instances.productService.sliding-window-size=10
resilience4j.circuitbreaker.instances.productService.minimum-number-of-calls=5
resilience4j.circuitbreaker.instances.productService.failure-rate-threshold=50
resilience4j.circuitbreaker.instances.productService.wait-duration-in-open-state=10s
resilience4j.circuitbreaker.instances.productService.permitted-number-of-calls-in-half-open-state=3
resilience4j.circuitbreaker.instances.productService.automatic-transition-from-open-to-half-open-enabled
```

sliding-window-type=COUNT_BASED
sliding-window-size=10
Tracks N (in this case 10) number of recent calls.

sliding-window-type=TIME_BASED
sliding-window-size=10s
Tracks calls made in a last N time duration(in this case 10sec)

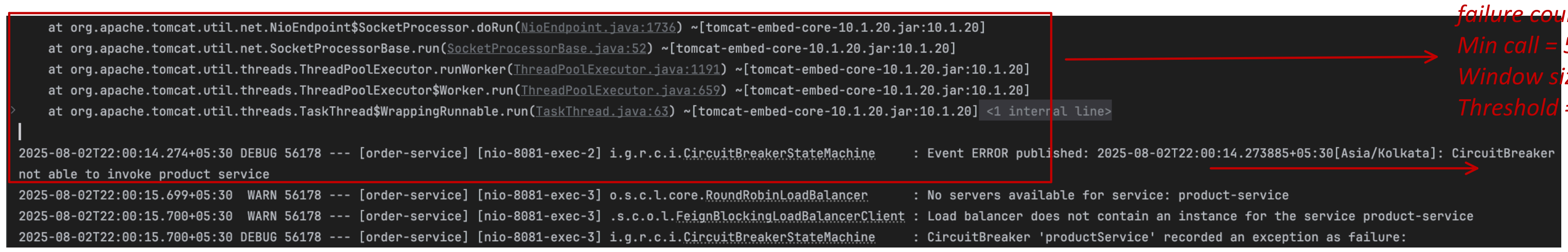
By default, the Circuit Breaker records all **RuntimeExceptions** and **Errors** as failures.

But if we want specific exception to be recorded and ignored, we can also configured it like below:



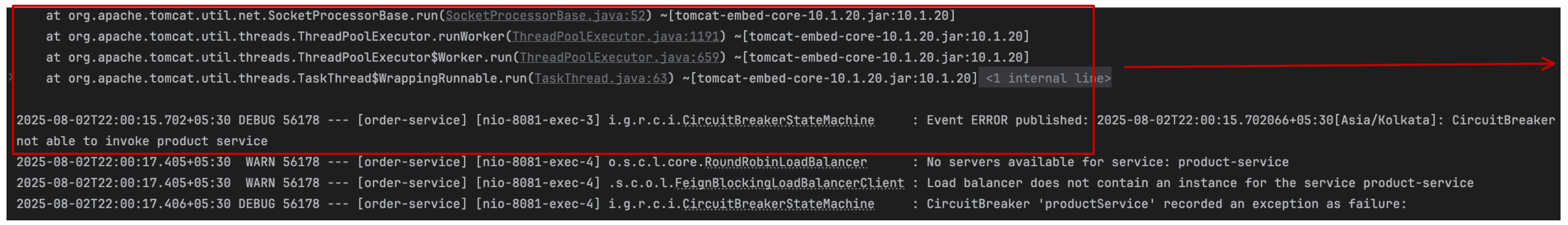
```
resilience4j.circuitbreaker.instances.productService.record-exceptions=java.io.IOException,org.springfr
resilience4j.circuitbreaker.instances.productService.ignore-exceptions=java.lang.IllegalArgumentExcepti
```

Output:



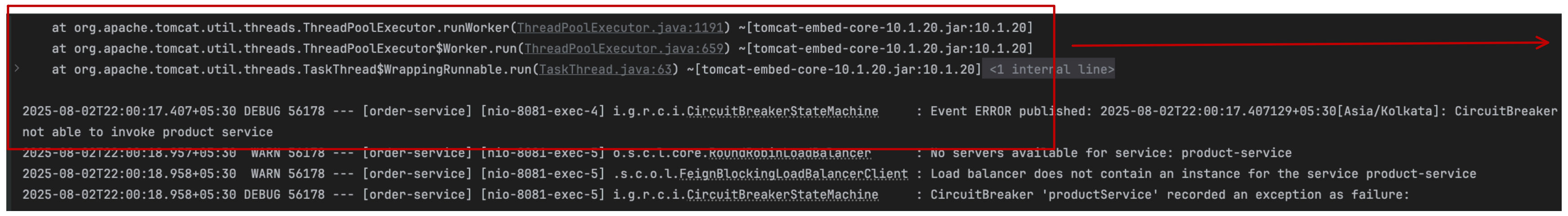
```
    at org.apache.tomcat.util.net.NioEndpoint$SocketProcessor.doRun(NioEndpoint.java:1736) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.net.SocketProcessorBase.run(SocketProcessorBase.java:52) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.threads.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1191) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.threads.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:659) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:63) ~[tomcat-embed-core-10.1.20.jar:10.1.20] <1 internal line>

2025-08-02T22:00:14.274+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-2] i.g.r.c.i.CircuitBreakerStateMachine     : Event ERROR published: 2025-08-02T22:00:14.273885+05:30[Asia/Kolkata]: CircuitBreaker
not able to invoke product service
2025-08-02T22:00:15.699+05:30  WARN 56178 --- [order-service] [nio-8081-exec-3] o.s.c.l.core.RoundRobinLoadBalancer      : No servers available for service: product-service
2025-08-02T22:00:15.700+05:30  WARN 56178 --- [order-service] [nio-8081-exec-3] .s.c.o.l.FeignBlockingLoadBalancerClient : Load balancer does not contain an instance for the service product-service
2025-08-02T22:00:15.700+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-3] i.g.r.c.i.CircuitBreakerStateMachine     : CircuitBreaker 'productService' recorded an exception as failure:
```

1st call to downstream : Failed
failure count: 1
Min call = 5
Window size = 10
Threshold = 50% of window size = 5 failure is the threshold



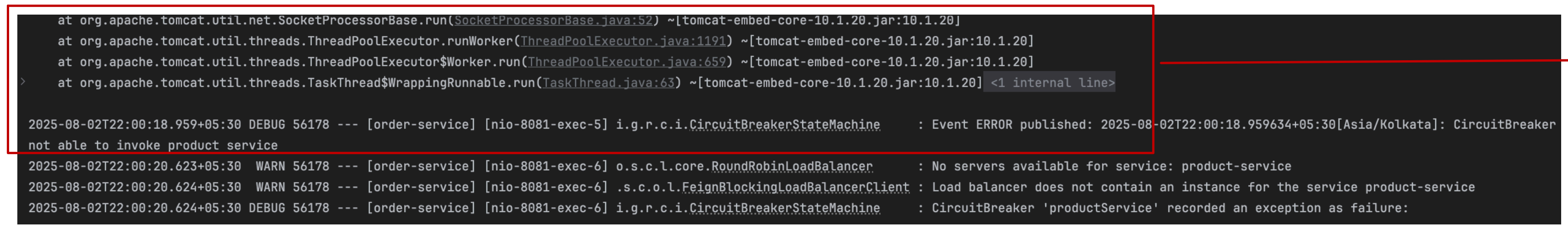
```
    at org.apache.tomcat.util.net.SocketProcessorBase.run(SocketProcessorBase.java:52) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.threads.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1191) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.threads.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:659) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:63) ~[tomcat-embed-core-10.1.20.jar:10.1.20] <1 internal line>

2025-08-02T22:00:15.702+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-3] i.g.r.c.i.CircuitBreakerStateMachine     : Event ERROR published: 2025-08-02T22:00:15.702066+05:30[Asia/Kolkata]: CircuitBreaker
not able to invoke product service
2025-08-02T22:00:17.405+05:30  WARN 56178 --- [order-service] [nio-8081-exec-4] o.s.c.l.core.RoundRobinLoadBalancer      : No servers available for service: product-service
2025-08-02T22:00:17.405+05:30  WARN 56178 --- [order-service] [nio-8081-exec-4] .s.c.o.l.FeignBlockingLoadBalancerClient : Load balancer does not contain an instance for the service product-service
2025-08-02T22:00:17.406+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-4] i.g.r.c.i.CircuitBreakerStateMachine     : CircuitBreaker 'productService' recorded an exception as failure:
```

2nd call to downstream : Failed
failure count: 2
Min call = 5
Window size = 10
Threshold = 50% of window size = 5 failure is the threshold



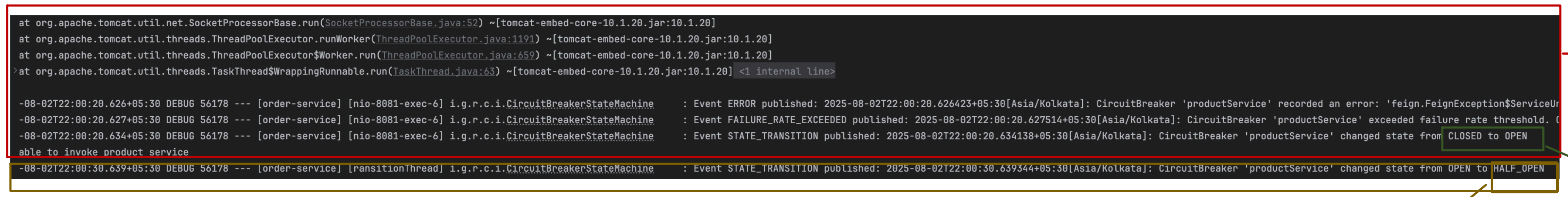
```
    at org.apache.tomcat.util.threads.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1191) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.threads.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:659) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:63) ~[tomcat-embed-core-10.1.20.jar:10.1.20] <1 internal line>

2025-08-02T22:00:17.407+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-4] i.g.r.c.i.CircuitBreakerStateMachine     : Event ERROR published: 2025-08-02T22:00:17.407129+05:30[Asia/Kolkata]: CircuitBreaker
not able to invoke product service
2025-08-02T22:00:18.957+05:30  WARN 56178 --- [order-service] [nio-8081-exec-5] o.s.c.l.core.RoundRobinLoadBalancer      : No servers available for service: product-service
2025-08-02T22:00:18.958+05:30  WARN 56178 --- [order-service] [nio-8081-exec-5] .s.c.o.l.FeignBlockingLoadBalancerClient : Load balancer does not contain an instance for the service product-service
2025-08-02T22:00:18.958+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-5] i.g.r.c.i.CircuitBreakerStateMachine     : CircuitBreaker 'productService' recorded an exception as failure:
```

3rd call to downstream : Failed
failure count: 3
Min call = 5
Window size = 10
Threshold = 50% of window size = 5 failure is the threshold



```
    at org.apache.tomcat.util.net.SocketProcessorBase.run(SocketProcessorBase.java:52) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.threads.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1191) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.threads.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:659) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:63) ~[tomcat-embed-core-10.1.20.jar:10.1.20] <1 internal line>

2025-08-02T22:00:18.959+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-5] i.g.r.c.i.CircuitBreakerStateMachine     : Event ERROR published: 2025-08-02T22:00:18.959634+05:30[Asia/Kolkata]: CircuitBreaker
not able to invoke product service
2025-08-02T22:00:20.623+05:30  WARN 56178 --- [order-service] [nio-8081-exec-6] o.s.c.l.core.RoundRobinLoadBalancer      : No servers available for service: product-service
2025-08-02T22:00:20.624+05:30  WARN 56178 --- [order-service] [nio-8081-exec-6] .s.c.o.l.FeignBlockingLoadBalancerClient : Load balancer does not contain an instance for the service product-service
2025-08-02T22:00:20.624+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-6] i.g.r.c.i.CircuitBreakerStateMachine     : CircuitBreaker 'productService' recorded an exception as failure:
```

4th call to downstream : Failed
failure count: 4
Min call = 5
Window size = 10
Threshold = 50% of window size = 5 failure is the threshold

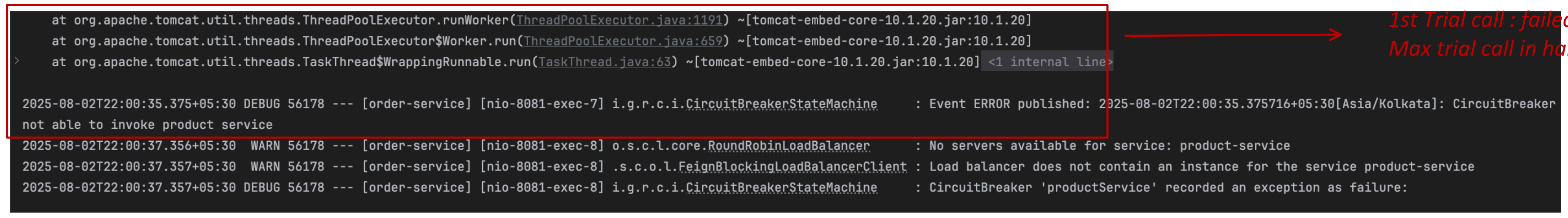


```
at org.apache.tomcat.util.net.SocketProcessorBase.run(SocketProcessorBase.java:52) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
at org.apache.tomcat.util.threads.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1191) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
at org.apache.tomcat.util.threads.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:659) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:63) ~[tomcat-embed-core-10.1.20.jar:10.1.20] <1 internal line>

-08-02T22:00:20.626+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-6] i.g.r.c.i.CircuitBreakerStateMachine     : Event ERROR published: 2025-08-02T22:00:20.626423+05:30[Asia/Kolkata]: CircuitBreaker 'productService' recorded an error: 'feign.FeignException$ServiceUr
-08-02T22:00:20.627+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-6] i.g.r.c.i.CircuitBreakerStateMachine     : Event FAILURE_RATE_EXCEEDED published: 2025-08-02T22:00:20.627514+05:30[Asia/Kolkata]: CircuitBreaker 'productService' exceeded failure rate threshold. (
-08-02T22:00:20.634+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-6] i.g.r.c.i.CircuitBreakerStateMachine     : Event STATE_TRANSITION published: 2025-08-02T22:00:20.634138+05:30[Asia/Kolkata]: CircuitBreaker 'productService' changed state from CLOSED to OPEN
able to invoke product service
-08-02T22:00:30.639+05:30 DEBUG 56178 --- [order-service] [ransitionThread] i.g.r.c.i.CircuitBreakerStateMachine     : Event STATE_TRANSITION published: 2025-08-02T22:00:30.639344+05:30[Asia/Kolkata]: CircuitBreaker 'productService' changed state from OPEN to HALF_OPEN
```

5th call to downstream : Failed
failure count: 5
Min call = 5
Window size = 10
Threshold = 50% of window size = 5 failure is the threshold

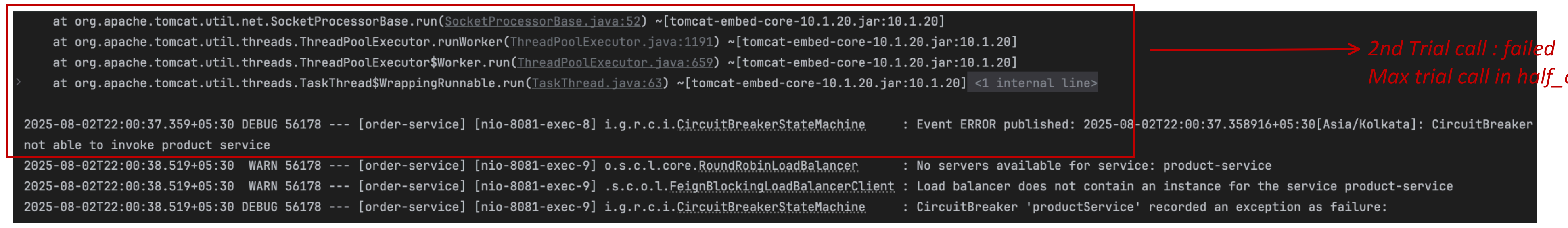
Now min call value is reached and also failure Threshold limit is touched, so state changed from CLOSED to OPEN

After 10sec of wait timeout,
state changed form OPEN to HALF_OPEN



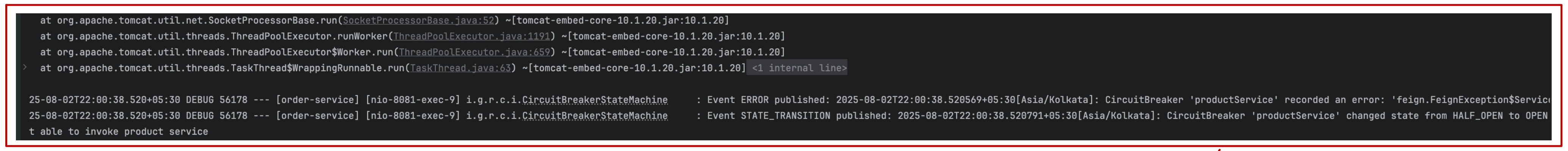
```
    at org.apache.tomcat.util.threads.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1191) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.threads.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:659) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:63) ~[tomcat-embed-core-10.1.20.jar:10.1.20] <1 internal line>

 2025-08-02T22:00:35.375+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-7] i.g.r.c.i.CircuitBreakerStateMachine     : Event ERROR published: 2025-08-02T22:00:35.375716+05:30[Asia/Kolkata]: CircuitBreaker
 not able to invoke product service
 2025-08-02T22:00:37.356+05:30  WARN 56178 --- [order-service] [nio-8081-exec-8] o.s.c.l.core.RoundRobinLoadBalancer      : No servers available for service: product-service
 2025-08-02T22:00:37.357+05:30  WARN 56178 --- [order-service] [nio-8081-exec-8] .s.c.o.l.FeignBlockingLoadBalancerClient : Load balancer does not contain an instance for the service product-service
 2025-08-02T22:00:37.357+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-8] i.g.r.c.i.CircuitBreakerStateMachine     : CircuitBreaker 'productService' recorded an exception as failure:
```

1st Trial call : failed
Max trial call in half_open state = 3



```
    at org.apache.tomcat.util.net.SocketProcessorBase.run(SocketProcessorBase.java:52) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.threads.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1191) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.threads.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:659) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
    at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:63) ~[tomcat-embed-core-10.1.20.jar:10.1.20] <1 internal line>

2025-08-02T22:00:37.359+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-8] i.g.r.c.i.CircuitBreakerStateMachine     : Event ERROR published: 2025-08-02T22:00:37.358916+05:30[Asia/Kolkata]: CircuitBreaker
not able to invoke product service
2025-08-02T22:00:38.519+05:30  WARN 56178 --- [order-service] [nio-8081-exec-9] o.s.c.l.core.RoundRobinLoadBalancer      : No servers available for service: product-service
2025-08-02T22:00:38.519+05:30  WARN 56178 --- [order-service] [nio-8081-exec-9] .s.c.o.l.FeignBlockingLoadBalancerClient : Load balancer does not contain an instance for the service product-service
2025-08-02T22:00:38.519+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-9] i.g.r.c.i.CircuitBreakerStateMachine     : CircuitBreaker 'productService' recorded an exception as failure:
```

2nd Trial call : failed
Max trial call in half_open state = 3



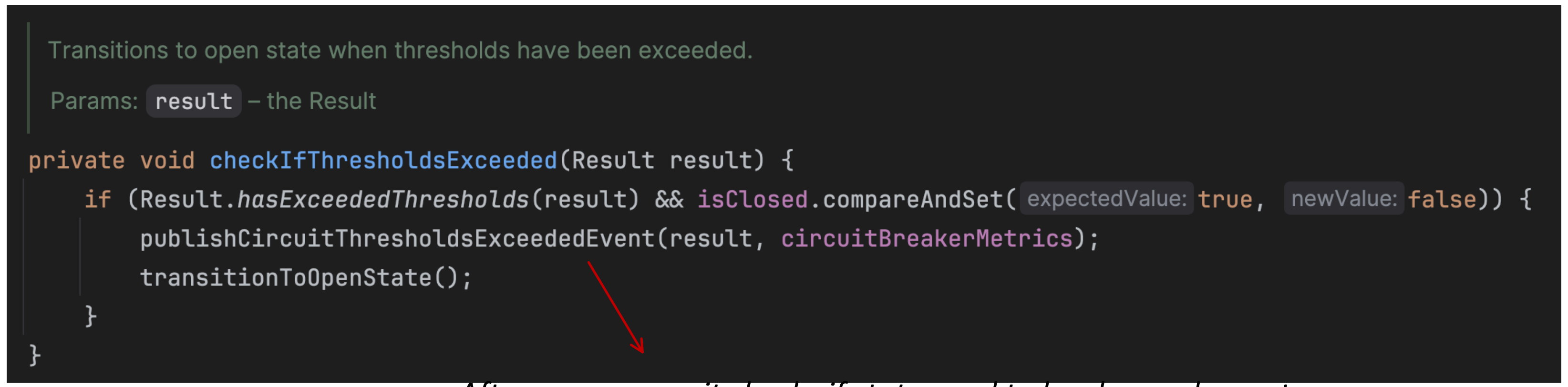
```
  at org.apache.tomcat.util.net.SocketProcessorBase.run(SocketProcessorBase.java:52) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
  at org.apache.tomcat.util.threads.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1191) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
  at org.apache.tomcat.util.threads.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:659) ~[tomcat-embed-core-10.1.20.jar:10.1.20]
  at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:63) ~[tomcat-embed-core-10.1.20.jar:10.1.20] <1 internal line>

25-08-02T22:00:38.520+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-9] i.g.r.c.i.CircuitBreakerStateMachine     : Event ERROR published: 2025-08-02T22:00:38.520569+05:30[Asia/Kolkata]: CircuitBreaker 'productService' recorded an error: 'feign.FeignException$Servic
25-08-02T22:00:38.520+05:30 DEBUG 56178 --- [order-service] [nio-8081-exec-9] i.g.r.c.i.CircuitBreakerStateMachine     : Event STATE_TRANSITION published: 2025-08-02T22:00:38.520791+05:30[Asia/Kolkata]: CircuitBreaker 'productService' changed state from HALF_OPEN to OPEN
t able to invoke product service
```

3rd Trial call : failed
Max trial call in half_open state = 3

Since max trial call limit reached, and success is not 100%, so status changed from HALF_OPEN to OPEN.

AOP intercept the call and pass it to "*CircuitBreakerStateMachine.java*" class.
Which has complete logic of changing one state to another, whenever there is a failure.

<u>Sample method from framework class:</u>



```
Transitions to open state when thresholds have been exceeded.
Params: result – the Result
private void checkIfThresholdsExceeded(Result result) {
    if (Result.hasExceededThresholds(result) && isClosed.compareAndSet( expectedValue: true,  newValue: false)) {
        publishCircuitThresholdsExceededEvent(result, circuitBreakerMetrics);
        transitionToOpenState();
    }
}
```

After every error, it checks if state need to be changed or not.
Like here its checking, if threshold limit reached and state is CLOSED, then transit to OPEN state.

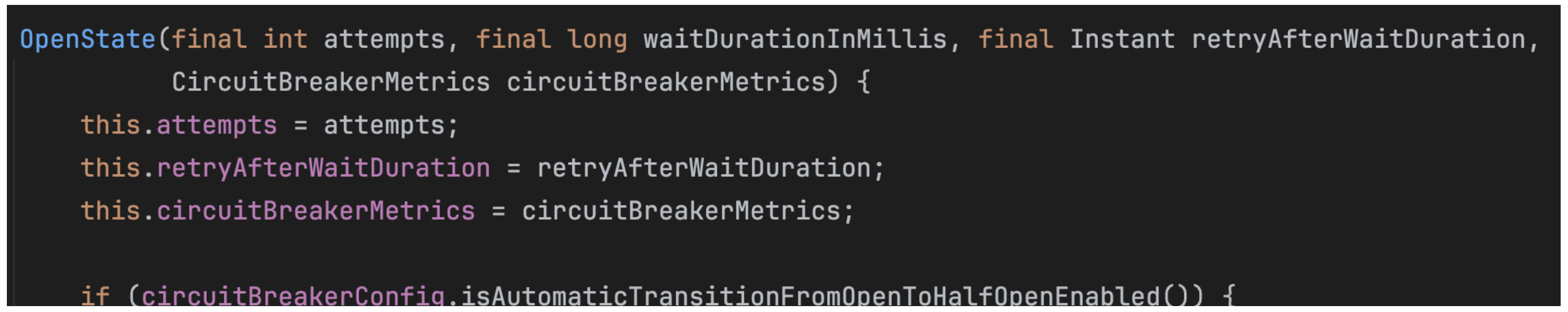
Likewise similar method is present for different state with specific transition logic.

One question, might be coming to you:

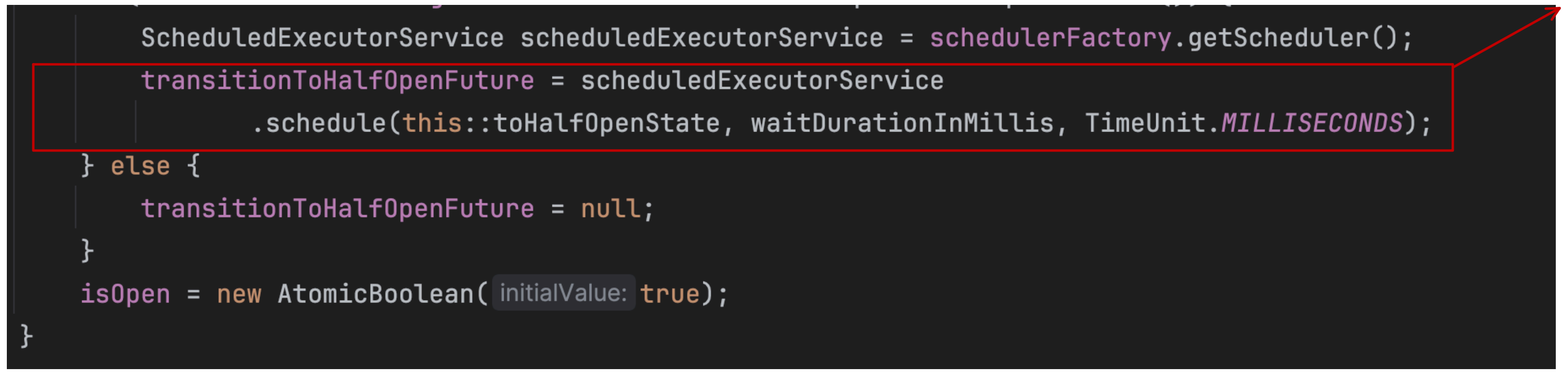
- Okay, for every failure call, we are checking if state need to be changed or not, make sense.
- But once it moved to OPEN state, then how automatically say after 10sec it move to HALF_OPEN state?

It uses, ScheduledThreadPoolExecutor

OpenState method from *CircuitBreakerStateMachine.java* framework class:



```
OpenState(final int attempts, final long waitDurationInMillis, final Instant retryAfterWaitDuration,
          CircuitBreakerMetrics circuitBreakerMetrics) {
    this.attempts = attempts;
    this.retryAfterWaitDuration = retryAfterWaitDuration;
    this.circuitBreakerMetrics = circuitBreakerMetrics;

    if (circuitBreakerConfig.isAutomaticTransitionFromOpenToHalfOpenEnabled()) {
```



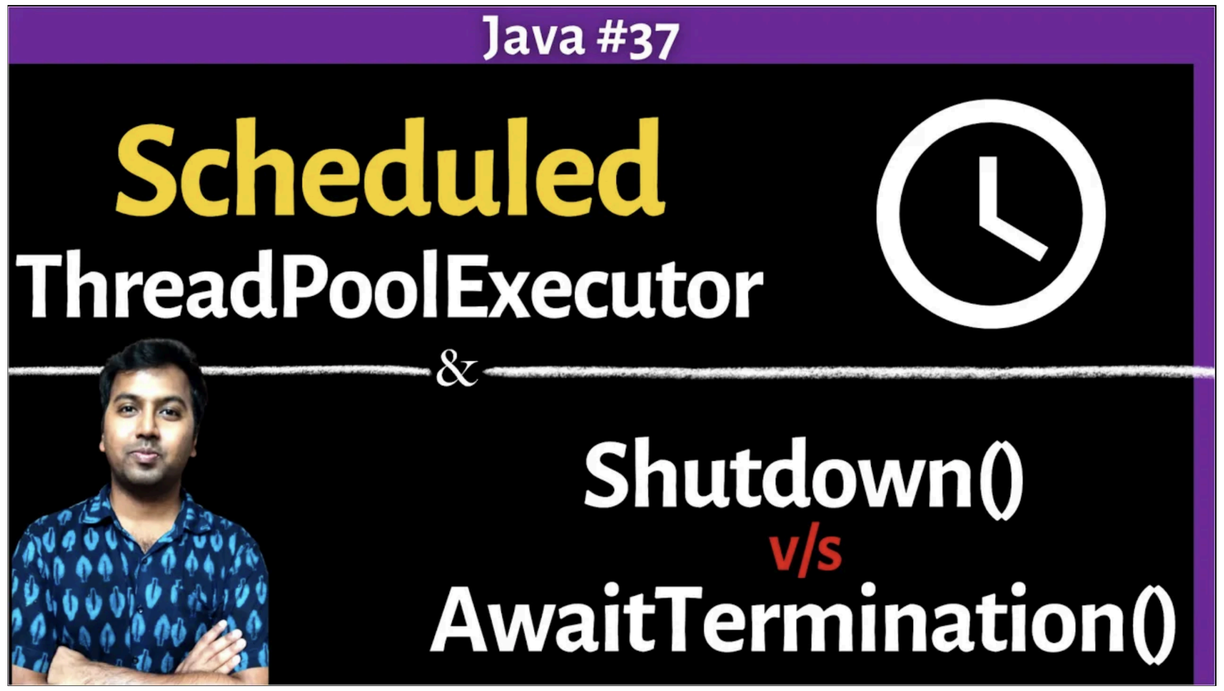
```
        ScheduledExecutorService scheduledExecutorService = schedulerFactory.getScheduler();
        transitionToHalfOpenFuture = scheduledExecutorService
                .schedule(this::toHalfOpenState, waitDurationInMillis, TimeUnit.MILLISECONDS);
    } else {
        transitionToHalfOpenFuture = null;
    }
    isOpen = new AtomicBoolean( initialValue: true);
}
```

It passes the request to ScheduledThreadPoolExecutor.
1st parameter is the Task i.e. "toHalfOpenState" to transit the state from OPEN to HALF_OPEN.

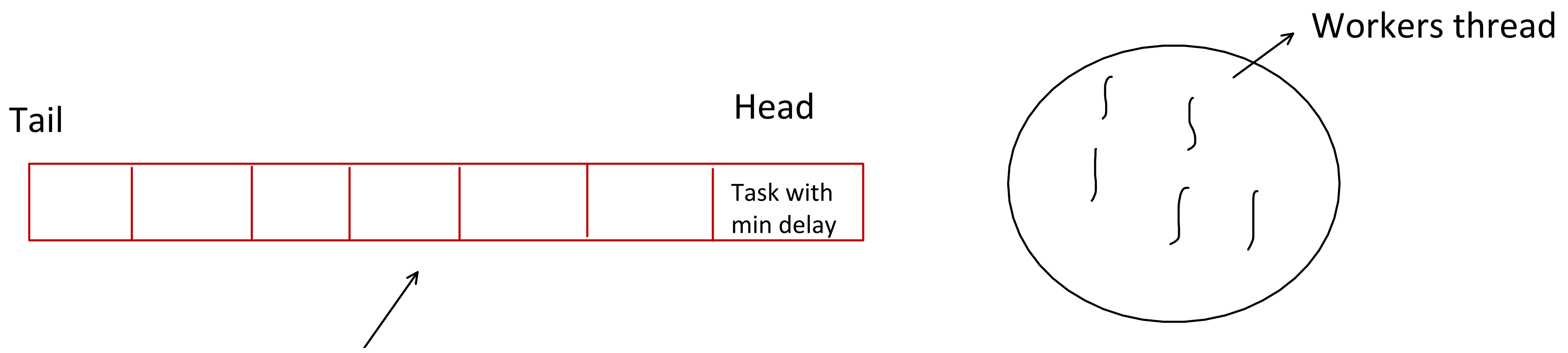
2nd parameter is the delay : like 10 or 20 or 30

3rd parameter is Time Unit: second or minutes or millisecond

I have already covered, how to use ScheduledThreadPoolExecutor in Java playlist. Pls check it out, if there is any doubt with its usage.



So internally **ScheduledThreadPoolExecutor** uses the concept of "DelayedQueue"



Priority queue, sorted based on min Delay

- Each available Worker thread, will look at the head of the Delayed Queue.

- If task delay is not yet expired, then thread waits(blocks) for that specific remaining delay period.
- Once the delay is over, OS wakes up the thread.
- One thread, pick the task from the Head and start executing it.
- Other thread start with the next task in the delayed queue, if task is not yet ready, then thread will wait(block) again.