NTT DATA

Resilencia en microservicios

Bootcamp Java Microservicios

NTTDATA

5-10-2022

Contenido

1. Herramientas Necesarias

- ✓ JDK Java 11
- ✓ IDE Java (Intellij IDEA 2022.1.2 Community, Spring Tool Suite).
- ✓ 2 Microservicios desarrollados previamente que se interconecten entre ambos

2. Caso de Uso

Se utilizará el mismo caso de uso del proyecto semanal

3. Instalando Resillence4j (vía POM)

- ✓ Modificar el archivo POM del microservicio product y agregar dependencias

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
	<modelVersion>4.0.0</modelVersion>
	<parent>
		<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-starter-parent</artifactId>
		<version>2.7.4</version>
		<relativePath/> <!-- lookup parent from repository -->
	</parent>
	<groupId>com.bootcamp.java</groupId>
	<artifactId>product</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<name>product</name>
	<description>Microservice for CRUD Product</description>
	<properties>
		<java.version>11</java.version>
		<org.mapstruct.version>1.5.2.Final</org.mapstruct.version>
		<lombok.version>1.18.24</lombok.version>
		<spring-cloud.version>2021.0.4</spring-cloud.version>
		<spring-boot-admin.version>2.7.4</spring-boot-admin.version>
	</properties>
	<dependencies>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-actuator</artifactId>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-data-mongodb-reactive</artifactId>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-webflux</artifactId>
		</dependency>

		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-devtools</artifactId>
```

```
            <scope>runtime</scope>
            <optional>true</optional>
        </dependency>
        <dependency>
            <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
            <optional>true</optional>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>io.projectreactor</groupId>
            <artifactId>reactor-test</artifactId>
            <scope>test</scope>
        </dependency>
        <dependency>
           <groupId>org.springframework.boot</groupId>
           <artifactId>spring-boot-starter-validation</artifactId>
      </dependency>
        <dependency>
            <groupId>org.mapstruct</groupId>
            <artifactId>mapstruct</artifactId>
            <version>${org.mapstruct.version}</version>
            <optional>true</optional>
        </dependency>
        <dependency>
            <groupId>org.mapstruct</groupId>
            <artifactId>mapstruct-processor</artifactId>
            <version>${org.mapstruct.version}</version>
        </dependency>
        <dependency>
          <groupId>org.springdoc</groupId>
          <artifactId>springdoc-openapi-webflux-core</artifactId>
          <version>1.4.3</version>
        </dependency>
        <dependency>
          <groupId>org.springdoc</groupId>
          <artifactId>springdoc-openapi-webflux-ui</artifactId>
          <version>1.4.3</version>
        </dependency>
        <dependency>
       <groupId>io.github.classgraph</groupId>
       <artifactId>classgraph</artifactId>
       <version>4.8.139</version>
    </dependency>
    <dependency>
            <groupId>org.springframework.cloud</groupId>
            <artifactId>spring-cloud-starter-config</artifactId>
        </dependency>
        <dependency>
           <groupId>de.codecentric</groupId>
           <artifactId>spring-boot-admin-starter-client</artifactId>
           <version>${spring-boot-admin.version}</version>
        </dependency>
```

```
            <dependency>
                  <groupId>org.springframework.cloud</groupId>
                  <artifactId>spring-cloud-starter</artifactId>
            </dependency>
              <dependency>
            <groupId>org.springframework.cloud</groupId>
            <artifactId>spring-cloud-starter-circuitbreaker-reactor-
resilience4j</artifactId>
        </dependency>
            <dependency>
               <groupId>org.springframework.boot</groupId>
               <artifactId>spring-boot-starter-aop</artifactId>
            </dependency>
      </dependencies>

      <build>
            <plugins>
                  <plugin>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-maven-plugin</artifactId>
                        <configuration>
                              <excludes>
                                    <exclude>
                                          <groupId>org.projectlombok</groupId>
                                          <artifactId>lombok</artifactId>
                                    </exclude>
                              </excludes>
                        </configuration>
                  </plugin>
                  <plugin>
                        <groupId>org.apache.maven.plugins</groupId>
                        <artifactId>maven-compiler-plugin</artifactId>
                        <version>3.5.1</version>
                        <configuration>
                              <source>${java.version}</source>
                              <target>${java.version}</target>
                              <annotationProcessorPaths>
                                  <path>
                                    <groupId>org.projectlombok</groupId>
                                      <artifactId>lombok</artifactId>
                                    <version>${lombok.version}</version>
                                 </path>
                                 <path>
                                      <groupId>org.mapstruct</groupId>
                                      <artifactId>mapstruct-processor</artifactId>
                                      <version>${org.mapstruct.version}</version>
                                 </path>
                                 <path>
                                      <groupId>org.projectlombok</groupId>
                                      <artifactId>lombok-mapstruct-binding</artifactId>
                                      <version>0.2.0</version>
                                  </path>
                              </annotationProcessorPaths>
                              <compilerArgs>
                                    <compilerArg>
                                          -Amapstruct.verbose=true
                                    </compilerArg>
```

```
                                </compilerArgs>
                        </configuration>
                </plugin>
        </plugins>
    </build>
    <dependencyManagement>
        <dependencies>
            <dependency>
                <groupId>org.springframework.cloud</groupId>
                <artifactId>spring-cloud-dependencies</artifactId>
                <version>${spring-cloud.version}</version>
                <type>pom</type>
                <scope>import</scope>
            </dependency>
        </dependencies>
    </dependencyManagement>
</project>
```

4. Circuit Breaker

- ✓ Modificar archivo .properties y agregar las siguientes líneas

```
resilience4j.circuitbreaker.instances.parameter-service.failureRateThreshold= 50
resilience4j.circuitbreaker.instances.parameter-service.minimumNumberOfCalls= 4
resilience4j.circuitbreaker.instances.parameter-service.slidingWindowType= COUNT_BASED
resilience4j.circuitbreaker.instances.parameter-service.slidingWindowSize= 8
resilience4j.circuitbreaker.instances.parameter-service.waitDurationInOpenState= 50s
resilience4j.circuitbreaker.instances.parameter-service.permittedNumberOfCallsInHalfOpenState= 3
```

- ✓ Ver opciones de configuración en: https://resilience4j.readme.io/docs/getting-started-3

5. Usando Resilience4J (Vía Código)

- ✓ Para hacer uso de circuit breaker por código, necesitamos hacer uso de la interfaz **ReactiveCircuitBreakerFactory**
- ✓ Crear una nueva instancia que ejecutara la acción original.
- ✓ En caso la acción falle se ejecutar a la acción alternativa

```
        @Autowired
        ReactiveCircuitBreakerFactory reactiveCircuitBreakerFactory;


        //Por codigo
        public Flux<Parameter> findById(String parameterId) {
                log.info("findById executed {}", parameterId);
                return webClient.get().uri("/v1/parameter/{parameterId}", parameterId).retrieve()
                                .onStatus(HttpStatus::is4xxClientError, clientResponse ->
Mono.error(new Exception("Error 400")))
                                .onStatus(HttpStatus::is5xxServerError, clientResponse ->
Mono.error(new Exception("Error 500")))
                                .bodyToFlux(Parameter.class)
                                .transform(it -> reactiveCircuitBreakerFactory.create("parameter-
service").run(it, throwable -> Flux.just(new Parameter())));
        }
```

6. Usando Resilience4J (Anotaciones)

- ✓ Para hacer uso de circuit breaker por código, necesitamos hacer uso de la interfaz la anotación **@CircuitBreaker**
- ✓ La cual creará una nueva instancia que ejecutará la acción original.
- ✓ En caso la acción falle se ejecutar a la acción alternativa (fallbackMethod)

```
       //Por anotaciones
       @CircuitBreaker(name = "parameter-service", fallbackMethod = "findByIdAlternative")
       public Flux<Parameter> findById(String parameterId) {
                log.info("findById executed {}", parameterId);
                return webClient.get().uri("/v1/parameter/{parameterId}",
parameterId).retrieve()
                                .onStatus(HttpStatus::is4xxClientError, clientResponse ->
Mono.error(new Exception("Error 400")))
                                .onStatus(HttpStatus::is5xxServerError, clientResponse ->
Mono.error(new Exception("Error 500")))
```

```
                        .bodyToFlux(Parameter.class);
    }

    public Flux<Parameter> findByIdAlternative(String parameterId, Exception ex) {
          log.info("findByIdAlternative executed {}", parameterId);
          log.error(ex.getMessage());
          return Flux.just(new Parameter());
    }
```

7. Mas información

- ✓ https://www.youtube.com/watch?v=LefFKbIFyjQ&t=210s&ab_channel=TechnoTownTechie
- ✓ https://www.youtube.com/watch?v=b6R4dElDtRc&ab_channel=JavaTechie