SpringCloud Sentinel整合Nacos实现配置持久化

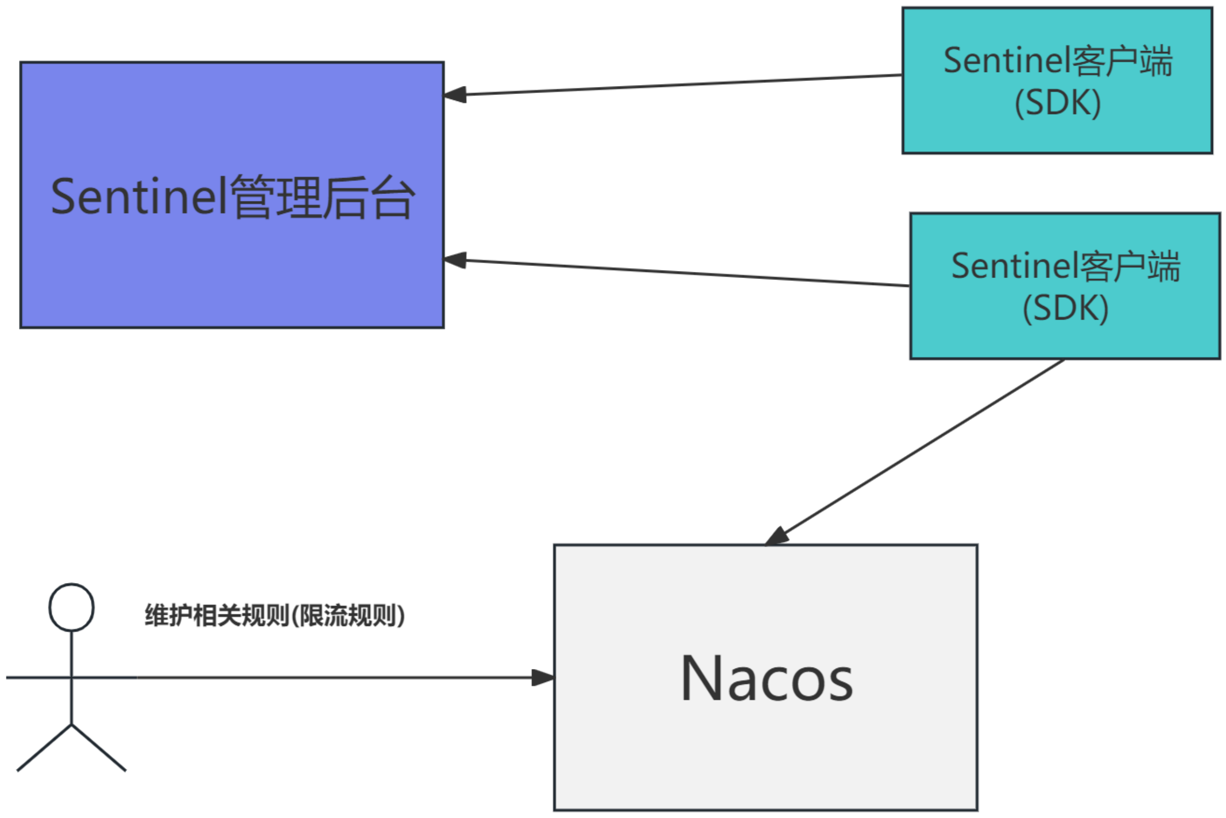
版本对照表

Spring Cloud Alibaba Version	Spring Cloud Version	Spring Boot Version
2021.0.5.0	Spring Cloud 2021.0.5	2.7.18

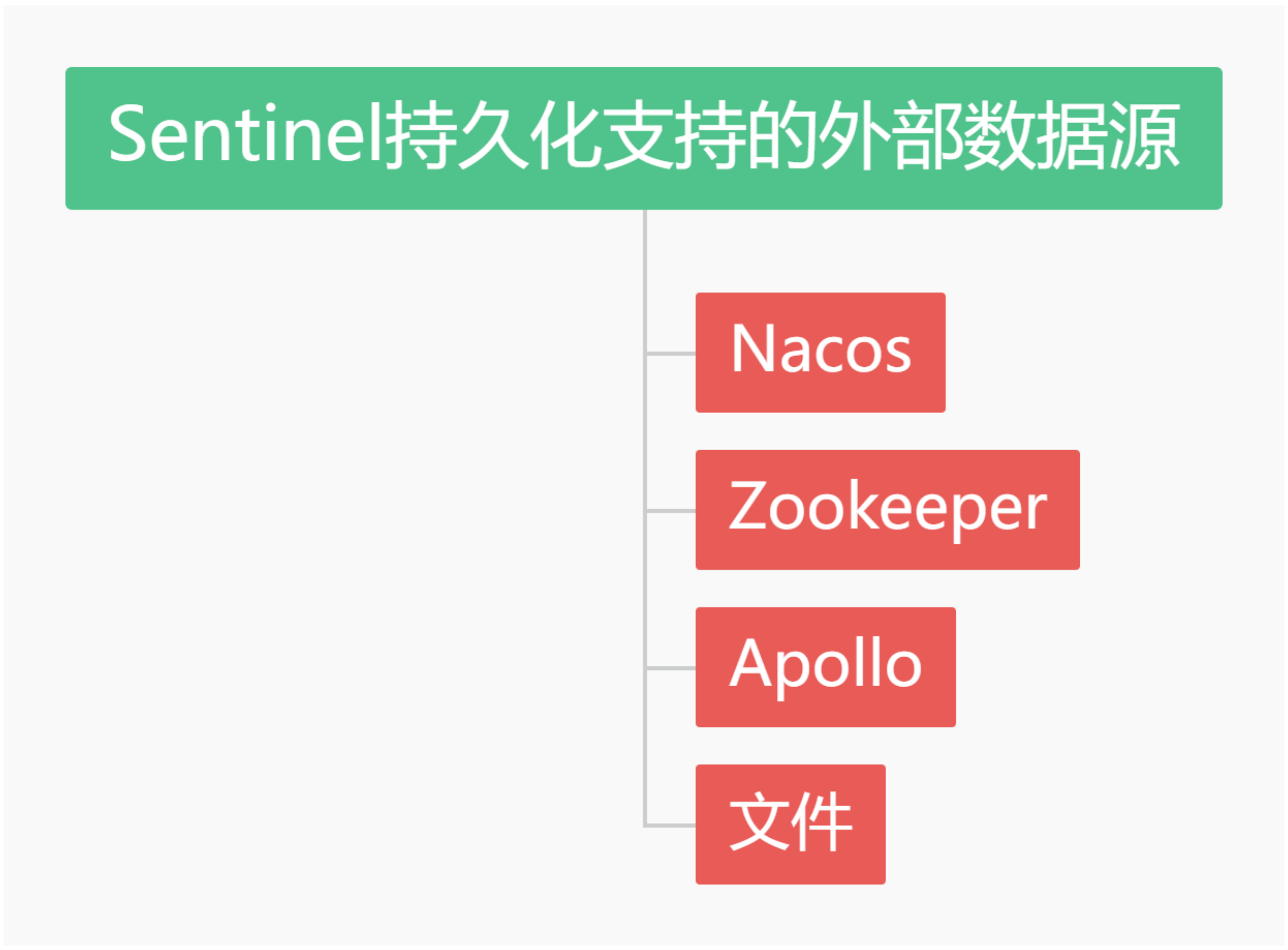
1 前言

Sentinel配置相关配置后无法持久化，服务重启之后就没了，所以整合nacos，在nacos服务持久化，Sentinel实时与nacos通信获取相关配置

Sentinel持久化限流规则到Nacos-单向同步



datasource:目前支持nacos、apollo、zookeeper、file 选择什么类型的数据源就配置相对应的key即可



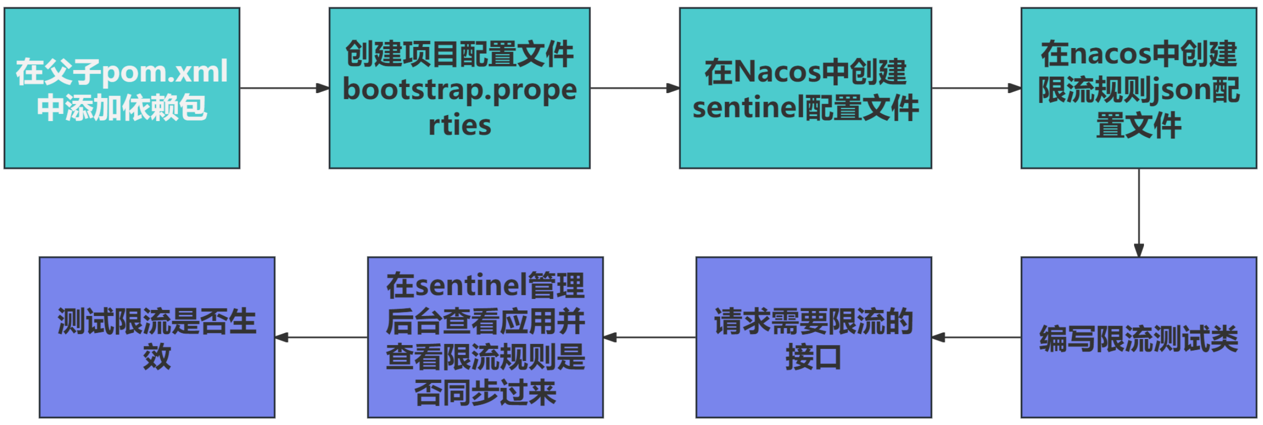
2 Sentinel配置数据源nacos

```
<dependency>
    <groupId>com.alibaba.csp</groupId>
    <artifactId>sentinel-datasource-
nacos</artifactId>
    <version>1.8.7</version>
</dependency>
```

3 Sentinel 启动

```
java -Dserver.port=8080 -
Dcsp.sentinel.dashboard.server=localhost:8080
-Dproject.name=sentinel-dashboard -jar
sentinel-dashboard-1.8.7.jar
```

4 整合流程



在父子pom.xml中添加依赖包
创建项目配置文件bootstrap.properties
在Nacos中创建sentinel配置文件
在nacos中创建限流规则json配置文件
编写限流测试类
请求需要限流的接口
在sentinel管理后台查看应用并查看限流规则是否同步过来
测试限流是否生效

5 SpringCloud Nacos 配置文件

bootstrap.properties

```
server.port=9090
spring.application.name=itbeien-spring-cloud-sentinel-nacos
spring.cloud.nacos.config.server-addr=127.0.0.1:3333
spring.cloud.nacos.config.username=nacos
spring.cloud.nacos.config.password=nacos
spring.cloud.nacos.config.file-extension=properties
spring.cloud.nacos.config.namespace=772c8d40-00a8-47dd-8eeb-dfe19fb76aa8
spring.cloud.nacos.config.group=dev
```

6 Sentinel配置文件

```
spring.application.name=itbeien-spring-cloud-sentinel-nacos
spring.cloud.nacos.discovery.server-addr=127.0.0.1:3333
spring.cloud.nacos.discovery.username=nacos
spring.cloud.nacos.discovery.password=nacos
# sentinel后台接口端口
spring.cloud.sentinel.transport.port=8719
spring.cloud.sentinel.transport.dashboard=127.0.0.1:8080

spring.cloud.sentinel.web-context-unify=false
spring.cloud.sentinel.datasource.flow-rules.nacos.server-addr=127.0.0.1:3333
spring.cloud.sentinel.datasource.flow-rules.nacos.namespace=772c8d40-00a8-47dd-8eeb-dfe19fb76aa8
spring.cloud.sentinel.datasource.flow-rules.nacos.username=nacos
spring.cloud.sentinel.datasource.flow-rules.nacos.password=nacos
spring.cloud.sentinel.datasource.flow-rules.nacos.dataId=${spring.application.name}-flow-rules
spring.cloud.sentinel.datasource.flow-rules.nacos.groupId=dev
spring.cloud.sentinel.datasource.flow-rules.nacos.data-type=json
spring.cloud.sentinel.datasource.flow-rules.nacos.rule-type=flow
```

7 Sentinel限流配置

```
[
    {
        "resource": "/api/testSentinel",
        "limitApp": "default",
        "grade": 1,
        "count": 2,
        "strategy": 0,
        "controlBehavior": 0,
        "clusterMode": false
    }
]
```

8 子模块pom.xml依赖

```
<?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/PO
M/4.0.0 http://maven.apache.org/xsd/maven-
4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <parent>
```

```
        <groupId>cn.itbeien</groupId>
        <artifactId>itbeien-
springcloudalibaba</artifactId>
        <version>1.0-SNAPSHOT</version>
    </parent>

    <artifactId>sentinel-nacos-
springcloud</artifactId>

    <dependencies>
        <dependency>

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

 <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-
web</artifactId>
        </dependency>
        <dependency>

 <groupId>com.alibaba.cloud</groupId>

            <artifactId>spring-cloud-starter-
```

```
alibaba-sentinel</artifactId>
        </dependency>
        <dependency>

 <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
        </dependency>
        <dependency>
            <groupId>com.alibaba.csp</groupId>
            <artifactId>sentinel-datasource-
nacos</artifactId>
        </dependency>
        <dependency>

 <groupId>com.alibaba.cloud</groupId>
            <artifactId>spring-cloud-starter-
alibaba-nacos-config</artifactId>
        </dependency>
        <dependency>

 <groupId>org.springframework.cloud</groupId>
            <artifactId>spring-cloud-starter-
bootstrap</artifactId>
        </dependency>
        <dependency>
 <groupId>com.fasterxml.jackson.core</groupId>
               <artifactId>jackson-
databind</artifactId>
        </dependency>
    </dependencies>
</project>
```

9 父模块pom.xml

```
<?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/PO
M/4.0.0 http://maven.apache.org/xsd/maven-
4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>cn.itbeien</groupId>
    <artifactId>itbeien-
springcloudalibaba</artifactId>
    <version>1.0-SNAPSHOT</version>
    <packaging>pom</packaging>
    <modules>
        <module>sentinel-nacos-
springcloud</module>
    </modules>
    <properties>
        <springboot-
version>2.7.18</springboot-version>
        <!--<springboot-
version>2.3.12.RELEASE</springboot-version>-->
        <springcloudalibaba-
version>2021.0.5.0</springcloudalibaba-
version>
        <spring-
cloud.version>2021.0.5</spring-cloud.version>
       <!-- <springcloudalibaba-
version>2021.1</springcloudalibaba-version>-->
        <lombok-version>1.18.30</lombok-
version>
        <sentinel-version>1.8.7</sentinel-
version>
       <!-- <nacos-
version>2.2.3.RELEASE</nacos-version>-->
        <jackson-databind-
version>2.17.0</jackson-databind-version>

    </properties>
    <dependencyManagement>
        <dependencies>
            <dependency>

 <groupId>org.springframework.boot</groupId>
                <artifactId>spring-
boot</artifactId>
                <version>${springboot-version}
</version>
            </dependency>
            <dependency>

 <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-
starter-web</artifactId>
                <version>${springboot-version}
</version>
            </dependency>
            <dependency>

 <groupId>com.alibaba.cloud</groupId>
                <artifactId>spring-cloud-
alibaba-dependencies</artifactId>
                <version>${springcloudalibaba-
version}</version>

                <type>pom</type>
                <scope>import</scope>
            </dependency>
            <dependency>

 <groupId>org.springframework.cloud</groupId>
                <artifactId>spring-cloud-
dependencies</artifactId>
                <version>${spring-
cloud.version}</version>
                <type>pom</type>
                <scope>import</scope>
            </dependency>
            <dependency>

 <groupId>org.projectlombok</groupId>

 <artifactId>lombok</artifactId>
                <version>${lombok-version}
</version>
            </dependency>
            <dependency>

 <groupId>com.alibaba.csp</groupId>
                <artifactId>sentinel-
datasource-nacos</artifactId>

                <version>${sentinel-version}
```

```
</version>
            </dependency>
            <dependency>

 <groupId>com.alibaba.cloud</groupId>
                <artifactId>spring-cloud-
starter-alibaba-nacos-config</artifactId>
                <version>${springcloudalibaba-
version}</version>
            </dependency>
            <dependency>

 <groupId>com.fasterxml.jackson.core</groupId>
                <artifactId>jackson-
databind</artifactId>
                <version>${jackson-databind-
version}</version>
            </dependency>
        </dependencies>

    </dependencyManagement>


</project>
```

10 Java代码

```
package cn.itbeien.controller;

import com.alibaba.csp.sentinel.annotation.SentinelResource;
import com.alibaba.csp.sentinel.slots.block.BlockException;
import lombok.extern.slf4j.Slf4j;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

/**
 * @author beien
 * @date 2024-04-02 20:38
 * Copyright© 2024 beien
 */
@RestController

@RequestMapping("api")
```

```
@Slf4j
public class SentinelController {

    @GetMapping("testSentinel")
    @SentinelResource(value =
"itbeienSentinel",fallback="handlerException")
    public String test() {
        log.info("sentinel流控测试...");
        return "success";
    }

    public  String handlerException(Throwable
exception) {
        return "请求太频繁，请稍后再试!";
    }

    //限流规则类
    public class ItbeienHandler {
        public  String
handlerExceptionOne(BlockException exception)
{
            return "10001, 服务不可用";
        }



        public  String
handlerExceptionTwo(BlockException exception)
{
                return "10002, 服务不可用";
            }


        }
}
```

11 配套视频

贝恩聊架构

扫描二维码，关注我的视频号