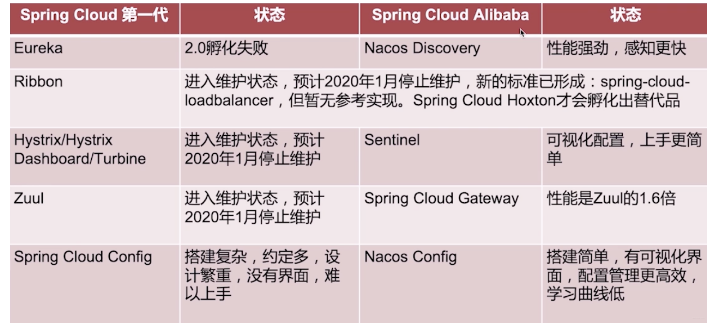
# 与spring cloud比较

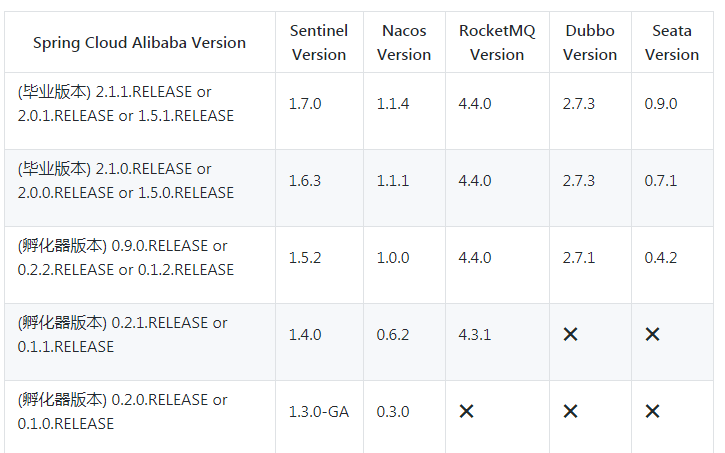


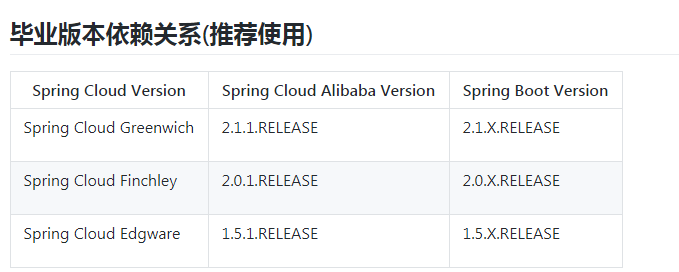
# Nacos

下载地址 <https://github.com/alibaba/nacos/releases>

解压启动,默认用户名密码为nacos,登录地址 <http://127.0.0.1:8848/nacos>

## 版本说明





## Maven依赖

<dependencyManagement>

<dependency>

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

<artifactId>spring-cloud-alibaba-dependencies</artifactId>

<version>2.1.1.RELEASE</version>

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencyManagement>

在dependencies中引入

<dependency>

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

<artifactId>spring-cloud-starter-alibaba-nacos-discovery</artifactId>

</dependency>

<dependency>

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

<artifactId>spring-cloud-starter-alibaba-nacos-config</artifactId>

</dependency>

## 注册中心

在配置文件application.properties中加入

#spring.application.name=alibaba-nacos-config-client:${spring.cloud.client.ip-address}:${server.port}

spring.application.name=alibaba-nacos-config-client2

server.port=8003

#server.servlet.context-path=/zuan

spring.cloud.nacos.discovery.server-addr=127.0.0.1:8848

## 配置中心

在bootstrap.properties中配置

spring.cloud.nacos.config.server-addr=127.0.0.1:8848

#配置文件dataid前置 默认${spring.application.name}

spring.cloud.nacos.config.prefix=alibaba-nacos-config-client

#配置文件dataid后置 默认properties

spring.cloud.nacos.config.file-extension=properties

#配置文件Group 默认DEFAULT\_GROUP

spring.cloud.nacos.config.group=DEFAULT\_GROUP

#spring.cloud.nacos.config.ext-config[0].data-id=name.properties

#spring.cloud.nacos.config.ext-config[0].group=DEFAULT\_GROUP

#spring.cloud.nacos.config.ext-config[0].refresh=true

#spring.cloud.nacos.config.ext-config[1].data-id=age.properties

#spring.cloud.nacos.config.ext-config[1].group=DEFAULT\_GROUP

#spring.cloud.nacos.config.ext-config[1].refresh=true

#共享配置,多个用逗号隔开 创建的dataid要包含后缀如name.properties

spring.cloud.nacos.config.shared-dataids=name.properties,age.properties

spring.cloud.nacos.config.refreshable-dataids=name.properties,age.properties

如果没有指定任何配置文件

默认dataid=${spring.application.name}

用法



# Dubbo

## Maven依赖

和引入nacos一样,先引入spring cloud Alibaba.

之后引入

<!-- Dubbo Spring Cloud Starter -->

<dependency>

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

<artifactId>spring-cloud-starter-dubbo</artifactId>

</dependency>

<!-- Spring Cloud Nacos Service Discovery -->

<dependency>

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

<artifactId>spring-cloud-starter-alibaba-nacos-discovery</artifactId>

</dependency>

## 配置

dubbo.scan.base-packages=com.yn.nacos.server.component.impl

dubbo.protocol.name=dubbo

dubbo.protocol.port=-1

dubbo.registry.address=spring-cloud://127.0.0.1

port=-1代表端口是递增的,从20880开始.

这是集成springcloud的所以以下配置也是必须的

spring.application.name=alibaba-nacos-discovery-server

server.port=8001

spring.cloud.nacos.discovery.server-addr=127.0.0.1:8848

之后的服务端和消费端就和普通的dubbo一样了,用service和refence

# Sentinel

## 下载启动

下载地址 <https://github.com/alibaba/Sentinel/releases>

启动 java -jar -Dserver.port=8888 sentinel-dashboard-1.7.0.jar

用户名密码：sentinel

* -Dsentinel.dashboard.auth.username=sentinel: 用于指定控制台的登录用户名为 sentinel；
* -Dsentinel.dashboard.auth.password=123456: 用于指定控制台的登录密码为 123456；如果省略这两个参数，默认用户和密码均为 sentinel
* -Dserver.servlet.session.timeout=7200: 用于指定 Spring Boot 服务端 session 的过期时间，如 7200 表示 7200 秒；60m 表示 60 分钟，默认为 30 分钟；

## MavenMaven引入

<dependency>

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

<artifactId>spring-cloud-starter-alibaba-sentinel</artifactId>

</dependency>

## 配置文件

spring.application.name=alibaba-sentinel-rate-limiting  
server.port=8001  
  
# sentinel dashboard  
spring.cloud.sentinel.transport.dashboard=localhost:8888

## 接入监控

Web服务自动接入,dubbo服务在方法加入注解@SentinelResource 即可



## 注意事项

项目引入的版本要与启动的sentinel服务版本一致,不然会监控不到数据.如我项目引入的是1.7.0,启动的是1.7.1,监控无数据,添加规则失败,改为启动1.7.0后成功.

# Sentinel存储nacos

## 引入maven依赖

<dependency>

<groupId>com.alibaba.csp</groupId>

<artifactId>sentinel-datasource-nacos</artifactId>

</dependency>

自动读取nacos的配置,所以需要

<dependency>

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

<artifactId>spring-cloud-starter-alibaba-nacos-config</artifactId>

</dependency>

## bootstrap.properties配置

#指定nacos地址

spring.cloud.nacos.config.server-addr=127.0.0.1:8848

#sentinel访问地址

spring.cloud.sentinel.transport.dashboard=localhost:8888

#sentinel连接的nacos地址

spring.cloud.sentinel.datasource.ds.nacos.server-addr=localhost:8848

#nacos中存储的dataId,与applitionname对应容易查找

spring.cloud.sentinel.datasource.ds.nacos.dataId=${spring.application.name}-sentinel

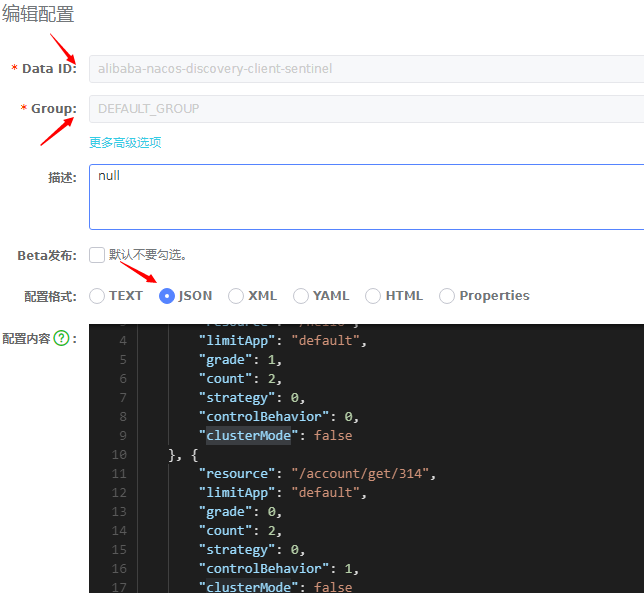
#nacos中规则对应groupId

spring.cloud.sentinel.datasource.ds.nacos.groupId=DEFAULT\_GROUP

#规则定义 对应类org.springframework.cloud.alibaba.sentinel.datasource.RuleType

spring.cloud.sentinel.datasource.ds.nacos.rule-type=flow

## nacos中创建限流规则



dataId与groupId和上面项目里配置的一致.

[  
 {  
 "resource": "/hello",  
 "limitApp": "default",  
 "grade": 1,  
 "count": 5,  
 "strategy": 0,  
 "controlBehavior": 0,  
 "clusterMode": false  
 }  
]

* resource：资源名，即限流规则的作用对象
* limitApp：流控针对的调用来源，若为 default 则不区分调用来源
* grade：限流阈值类型（QPS 或并发线程数）；0代表根据并发数量来限流，1代表根据QPS来进行流量控制
* count：限流阈值
* strategy：调用关系限流策略
* controlBehavior：流量控制效果（直接拒绝、Warm Up、匀速排队）
* clusterMode：是否为集群模式

启动程序后就能看到sentinel多了两条限流规则.



注意

* Sentinel控制台中修改规则：仅存在于服务的内存中，不会修改Nacos中的配置值，重启后恢复原来的值。
* Nacos控制台中修改规则：服务的内存中规则会更新，Nacos中持久化规则也会更新，重启后依然保持。

# Geteaway