链滴

Spring Cloud Gateway- 过滤器工厂详解（GatewayFilter Factories）

作者：jianzh5

原文链接：https://ld246.com/article/1579224438713

来源网站：链滴

这一节来探讨Spring Cloud Gateway内置的Filter工厂。包括：

- AddRequestHeader GatewayFilter Factory
- AddRequestParameter GatewayFilter Factory
- AddResponseHeader GatewayFilter Factory
- DedupeResponseHeader GatewayFilter Factory
- Hystrix GatewayFilter Factory
- FallbackHeaders GatewayFilter Factory
- PrefixPath GatewayFilter Factory
- PreserveHostHeader GatewayFilter Factory
- RequestRateLimiter GatewayFilter Factory
- RedirectTo GatewayFilter Factory
- RemoveHopByHopHeadersFilter GatewayFilter Factory
- RemoveRequestHeader GatewayFilter Factory
- RemoveResponseHeader GatewayFilter Factory
- RewritePath GatewayFilter Factory
- RewriteResponseHeader GatewayFilter Factory
- SaveSession GatewayFilter Factory
- SecureHeaders GatewayFilter Factory
- SetPath GatewayFilter Factory
- SetResponseHeader GatewayFilter Factory
- SetStatus GatewayFilter Factory
- StripPrefix GatewayFilter Factory
- Retry GatewayFilter Factory
- RequestSize GatewayFilter Factory
- Modify Request Body GatewayFilter Factory
- Modify Response Body GatewayFilter Factory
- Default Filters

技巧

断点打在 org.springframework.cloud.gateway.filter.NettyRoutingFilter#filter ， 就可以调试Gateay转发的具体细节了。

添加如下配置，可观察到一些请求细节：

```
logging:
  level:
    org.springframework.cloud.gateway: trace
    org.springframework.http.server.reactive: debug
    org.springframework.web.reactive: debug
    reactor.ipc.netty: debug
```

1 AddRequestHeader GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: add_request_header_route
        uri: http://javadaily.cn
        filters:
        - AddRequestHeader=X-Request-Foo, Bar
```

为原始请求添加名为 X-Request-Foo ，值为 Bar 的请求头。

2 AddRequestParameter GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: add_request_parameter_route
        uri: http://javadaily.com
        filters:
        - AddRequestParameter=foo, bar
```

为原始请求添加请求参数 foo=bar

3 AddResponseHeader GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: add_response_header_route
        uri: http://javadaily.cn
        filters:
        - AddResponseHeader=X-Response-Foo, Bar
```

添加名为 X-Request-Foo ，值为 Bar 的响应头。

4 DedupeResponseHeader GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: dedupe_response_header_route
        uri: http://javadaily.cn
        filters:
        - DedupeResponseHeader=Access-Control-Allow-Credentials Access-Control-Allow-Orig
n, RETAIN_FIRST
```

剔除重复的响应头。

举个例子：

我们在Gateway以及微服务上都设置了CORS（解决跨域）header，如果不做任何配置，请求 -> 网关 -> 微服务，获得的响应就是这样的：

Access-Control-Allow-Credentials: true, true

Access-Control-Allow-Origin: https://musk.mars, https://musk.mars

也就是Header重复了。要想把这两个Header去重，只需设置成如下即可。

filters:

● DedupeResponseHeader=Access-Control-Allow-Credentials Access-Control-Allow-Origin

也就是说，想要去重的Header如果有多个，用空格分隔即可；

去重策略：

RETAIN_FIRST: 默认值，保留第一个值

RETAIN_LAST: 保留最后一个值

RETAIN_UNIQUE: 保留所有唯一值，以它们第一次出现的顺序保留

5 Hystrix GatewayFilter Factory

Hystrix是Spring Cloud第一代中的容错组件，不过已经进入维护模式，取而代之的是Alibaba Sentine /Resilience4J。

所以本文不做详细探讨了，但Gateway整合Hystrix其实包含了很多姿势。请感兴趣的同学自行前往方文档了解详情：https://cloud.spring.io/spring-cloud-static/Greenwich.SR2/single/spring-clou .html#hystrix 。

```yml
spring:
  cloud:
    gateway:
      routes:
      - id: hystrix_route
        uri: http://javadaily.cn
        filters:
        - Hystrix=myCommandName
```

6 FallbackHeaders GatewayFilter Factory

也是对Hystrix的支持，不做详细探讨了，请感兴趣的同学自行前往官方文档了解详情：https://cloud. pring.io/spring-cloud-static/Greenwich.SR2/single/spring-cloud.html#fallback-headers

```yml
spring:
  cloud:
    gateway:
      routes:
      - id: ingredients
        uri: lb://ingredients
        predicates:
        - Path=//ingredients/**
```

```
        filters:
        - name: Hystrix
          args:
            name: fetchIngredients
            fallbackUri: forward:/fallback
      - id: ingredients-fallback
        uri: http://localhost:9994
        predicates:
        - Path=/fallback
        filters:
        - name: FallbackHeaders
          args:
            executionExceptionTypeHeaderName: Test-Header
```

7 PrefixPath GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: prefixpath_route
        uri: http://javadaily.cn
        filters:
        - PrefixPath=/mypath
```

为匹配的路由添加前缀。例如：访问${GATEWAY_URL}/hello 会转发到http://javadaily.com/mypat/hello 。

8 PreserveHostHeader GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: preserve_host_route
        uri: http://javadaily.cn
        filters:
        - PreserveHostHeader
```

如果不设置，那么名为 Host 的Header由Http Client控制；如果设置了，那么会设置一个请求属性（reserveHostHeader=true），路由过滤器会检查从而去判断是否要发送原始的、名为Host的Heade
。

9 RequestRateLimiter GatewayFilter Factory

基于redis的限流过滤器工厂

```
spring:
  cloud:
    gateway:
      routes:
      - id: requestratelimiter_route
        uri: http://javadaily.cn
        filters:
```

```
      - name: RequestRateLimiter
        args:
          redis-rate-limiter.replenishRate: 10
          redis-rate-limiter.burstCapacity: 20
```

10 RedirectTo GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: prefixpath_route
        uri: http://javadaily.cn
        filters:
        # 配置成HTTP状态码, URL的形式
        - RedirectTo=302, http://www.itmuch.com
```

HTTP状态码应该是HTTP状态码300序列，例如301

URL必须是合法的URL，并且该值会作为名为 Location 的Header。

上面配置表达的意思是： ${GATEWAY_URL}/hello 会重定向到 http://javadaily.cn/hello ，并且携一个 Location:http://www.itmuch.com 的Header。

11 RemoveHopByHopHeadersFilter GatewayFilter Factory

spring.cloud.gateway.filter.remove-hop-by-hop.headers: Connection,Keep-Alive

移除转发请求的Header，多个用 , 分隔。默认情况下，移除如下Header。这些Header是由 IETF 组规定的。

- Connection
- Keep-Alive
- Proxy-Authenticate
- Proxy-Authorization
- TE
- Trailer
- Transfer-Encoding
- Upgrade

12 RemoveRequestHeader GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: removerequestheader_route
        uri: http://javadaily.cn
        filters:
        - RemoveRequestHeader=X-Request-Foo
```

为原始请求删除名为 X-Request-Foo 的请求头。

13 RemoveResponseHeader GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: removeresponseheader_route
        uri: http://javadaily.cn
        filters:
        - RemoveResponseHeader=X-Response-Foo
```

删除名为 X-Request-Foo 的响应头。

14 RewritePath GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: rewritepath_route
        uri: http://javadaily.cn
        predicates:
        - Path=/foo/**
        filters:
        # 配置成原始路径正则, 重写后的路径的正则
        - RewritePath=/foo/(?<segment>.*), /$\{segment}
```

重写请求路径。如上配置，访问 /foo/bar 会将路径改为/bar 再转发，也就是会转发到 http://javadaily.cn/bar 。需要注意的是，由于YAML语法，需用\ 替换 。

15 RewriteResponseHeader GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: rewriteresponseheader_route
        uri: http://javadaily.cn
        filters:
        - RewriteResponseHeader=X-Response-Foo, password=[^&]+, password=***
```

如果名为 X-Response-Foo 的响应头的内容是/42? user=ford&password=omg!what&flag=true 则会被修改为/42?user=ford&password=***&flag=true。

16 SaveSession GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
```

```
- id: save_session
  uri: http://javadaily.cn
  predicates:
  - Path=/foo/**
  filters:
  - SaveSession
```

在转发到后端微服务请求之前，强制执行 WebSession::save 操作。用在那种像 Spring Session 延数据存储（笔者注：数据不是立刻持久化）的，并希望在请求转发前确保session状态保存情况。

如果你将Spring Secutiry于Spring Session集成使用，并想确保安全信息都传到下游机器，你就需要置这个filter。

17 SecureHeaders GatewayFilter Factory

添加一系列起安全作用的响应头。Spring Cloud Gateway参考了这篇博客的建议：https://blog.appanary.com/2017/http-security-headers.html

默认会添加如下Header（包括值）：

- X-Xss-Protection:1; mode=block
- Strict-Transport-Security:max-age=631138519
- X-Frame-Options:DENY
- X-Content-Type-Options:nosniff
- Referrer-Policy:no-referrer
- Content-Security-Policy:default-src 'self' https:; font-src 'self' https: data:; img-src 'self' https: data:; object-src 'none'; script-src https:; style-src 'self' https: 'unsafe-inline'
- X-Download-Options:noopen
- X-Permitted-Cross-Domain-Policies:none

如果你想修改这些Header的值，可使用如下配置：

前缀：spring.cloud.gateway.filter.secure-headers

上面的header对应的后缀：

- xss-protection-header
- strict-transport-security
- frame-options
- content-type-options
- referrer-policy
- content-security-policy
- download-options
- permitted-cross-domain-policies

例如：spring.cloud.gateway.filter.secure-headers.xss-protection-header: 你想要的值

如果想禁用某些Header，可使用如下配置：spring.cloud.gateway.filter.secure-headers.disable

多个用 , 分隔。例如：spring.cloud.gateway.filter.secure-headers.disable=frame-options,download-options 。

18 SetPath GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: setpath_route
        uri: http://javadaily.cn
        predicates:
        - Path=/foo/{segment}
        filters:
        - SetPath=/{segment}
```

采用路径template参数，通过请求路径的片段的模板化，来达到操作修改路径的母的，运行多个路径段模板化。

如上配置，访问${GATEWAY_PATH}/foo/bar ，则对于后端微服务的路径会修改为 /bar 。

19 SetResponseHeader GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: setresponseheader_route
        uri: http://example.org
        filters:
        - SetResponseHeader=X-Response-Foo, Bar
```

如果后端服务响应带有名为 X-Response-Foo 的响应头，则将值改为替换成 Bar 。

20 SetStatus GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: setstatusstring_route
        uri: http://example.org
        filters:
        - SetStatus=BAD_REQUEST
      - id: setstatusint_route
        uri: http://example.org
        filters:
        - SetStatus=401
```

修改响应的状态码，值可以是数字，也可以是字符串。但一定要是Spring HttpStatus 枚举类中的值
如上配置，两种方式都可以返回HTTP状态码401。

21 StripPrefix GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: nameRoot
        uri: http://nameservice
        predicates:
        - Path=/name/**
        filters:
        - StripPrefix=2
```

数字表示要截断的路径的数量。如上配置，如果请求的路径为 /name/bar/foo ，则路径会修改为/foo ，也就是会截断2个路径。

22 Retry GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: retry_test
        uri: http://localhost:8080/flakey
        predicates:
        - Host=*.retry.com
        filters:
        - name: Retry
          args:
            retries: 3
            statuses: BAD_GATEWAY
```

针对不同的响应做重试，可配置如下参数：

- retries: 重试次数
- statuses: 需要重试的状态码，取值在 org.springframework.http.HttpStatus 中
- methods: 需要重试的请求方法，取值在 org.springframework.http.HttpMethod 中
- series: HTTP状态码系列，取值在 org.springframework.http.HttpStatus.Series 中

23 RequestSize GatewayFilter Factory

```
spring:
  cloud:
    gateway:
      routes:
      - id: request_size_route
      uri: http://localhost:8080/upload
      predicates:
      - Path=/upload
      filters:
      - name: RequestSize
```

```
      args:
        # 单位字节
        maxSize: 5000000
```

为后端服务设置收到的最大请求包大小。如果请求大小超过设置的值，则返回 413 Payload Too Larg 。默认值是5M

24 Modify Request Body GatewayFilter Factory

TIPS

该过滤器处于 BETA 状态，未来API可能会变化，生产环境请慎用。

可用于在Gateway将请求发送给后端微服务之前，修改请求体内容。该过滤器只能通过代码配置，不持在配置文件设置。示例：

```
@Bean
public RouteLocator routes(RouteLocatorBuilder builder) {
    return builder.routes()
        .route("rewrite_request_obj", r -> r.host("*.rewriterequestobj.org")
            .filters(f -> f.prefixPath("/httpbin")
                .modifyRequestBody(String.class, Hello.class, MediaType.APPLICATION_JSON_VALU
,
                    (exchange, s) -> return Mono.just(new Hello(s.toUpperCase()))))).uri(uri))
        .build();
}

static class Hello {
    String message;

    public Hello() { }

    public Hello(String message) {
        this.message = message;
    }

    public String getMessage() {
        return message;
    }

    public void setMessage(String message) {
        this.message = message;
    }
}
```

25 Modify Response Body GatewayFilter Factory

TIPS

该过滤器处于 BETA 状态，未来API可能会变化，生产环境请慎用。

可用于修改响应体内容。该过滤器只能通过代码配置，不支持在配置文件设置。示例：

```
@Bean
```

```
public RouteLocator routes(RouteLocatorBuilder builder) {
    return builder.routes()
        .route("rewrite_response_upper", r -> r.host("*.rewriteresponseupper.org")
            .filters(f -> f.prefixPath("/httpbin")
                .modifyResponseBody(String.class, String.class,
                    (exchange, s) -> Mono.just(s.toUpperCase()))).uri(uri)
        .build();
}
```

26 Default Filters

```
spring:
  cloud:
    gateway:
      default-filters:
      - AddResponseHeader=X-Response-Default-Foo, Default-Bar
      - PrefixPath=/httpbin
```

如果你想为所有路由添加过滤器，可使用该属性。