API Gateway是为了系统提供统一的接入网关机构，Spring Cloud Gateway 是 Spring 官方基于 Spring 5.0，Spring Boot 2.0 和 Project Reactor 等技术开发的网关，Spring Cloud Gateway 旨在为微服务架构提供一种简单而有效的统一的 API 路由管理方式。**Spring Cloud Gateway 作为 Spring Cloud 生态系中的网关，目标是替代 Netflix ZUUL**，其不仅提供统一的路由方式，并且基于 Filter 链的方式提供了网关基本的功能，例如：安全，监控/埋点，和限流等。

Spring Cloud Gateway的工程流程如下图所示：



客户端向 Spring Cloud Gateway 发出请求。然后在 Gateway Handler Mapping 中找到与请求相匹配的路由，将其发送到 Gateway Web Handler。Handler 再通过指定的过滤器链来将请求发送到我们实际的服务执行业务逻辑，然后返回。过滤器之间用虚线分开是因为过滤器可能会在发送代理请求之前（pre）或之后（post）执行业务逻辑。

值得注意的是，Spring Cloud Gateway 不使用 Web 作为服务器，而是 使用 WebFlux 作为服务器，Gateway 项目已经依赖了 starter-webflux，所以这里 不要依赖 starter-web。

Spring Cloud Gateway的application.yml配置具体如下：

|  |
| --- |
| 服务中的nacos注册 |
| spring:  application:  name: ethereum-provider  cloud:  nacos:  discovery:  server-addr: 127.0.0.1:8848  spring:  application:  name: spring-gateway  cloud:  nacos:  discovery:  server-addr: 127.0.0.1:8848  sentinel:  transport:  port: 8721  dashboard: localhost:8080  gateway:  discovery:  locator:  enabled: true  routes:  - id: NACOS-CONSUMER  uri: lb://nacos-consumer  predicates:  - Method=GET,POST  - id: NACOS-CONSUMER-FEIGN  uri: lb://nacos-consumer-feign  predicates:  - Method=GET,POST |

API Gateway的Application类上添加注解@EnableFeignClients，具体如下：

|  |
| --- |
| 添加@ EnableFeignClients的Application |
| @SpringBootApplication  @EnableDiscoveryClient  @EnableFeignClients  public class GatewayApplication {  public static void main(String[] args) {  SpringApplication.run(GatewayApplication.class, args);  }  } |