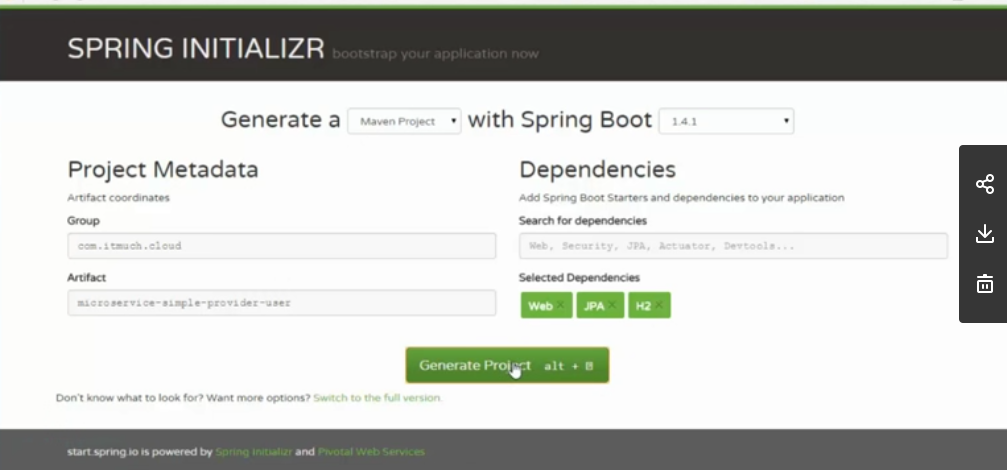
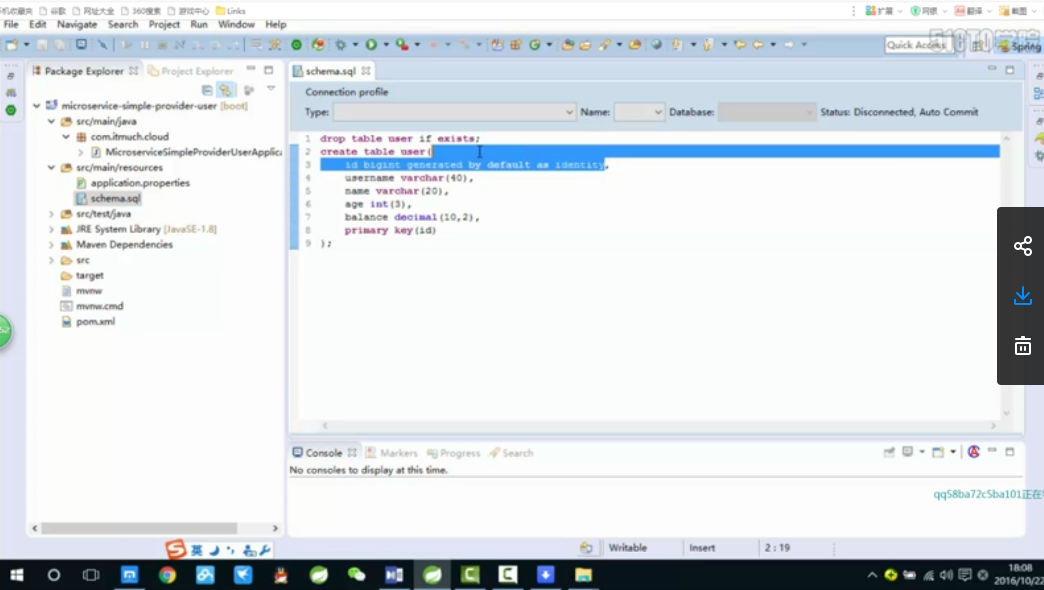
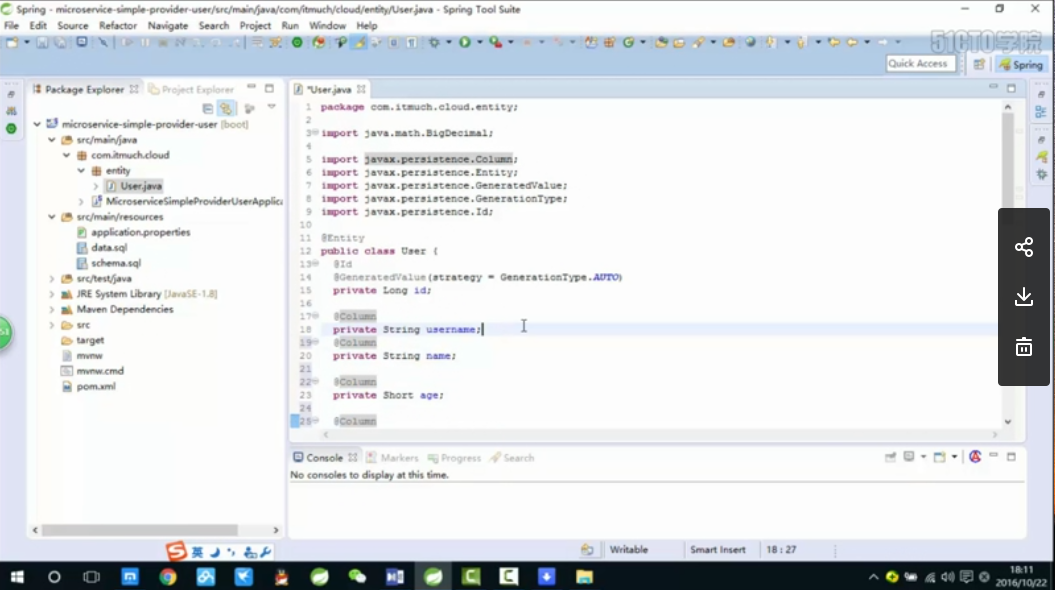
创建文件



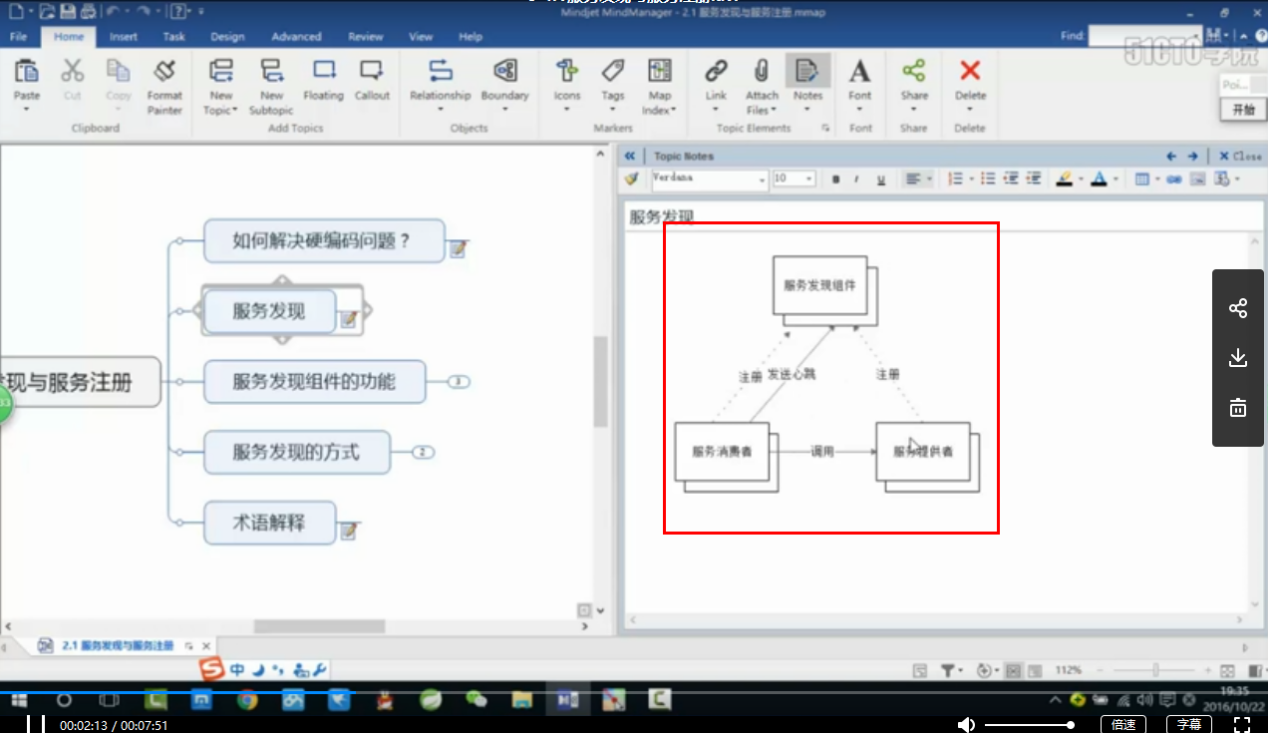
建表语句

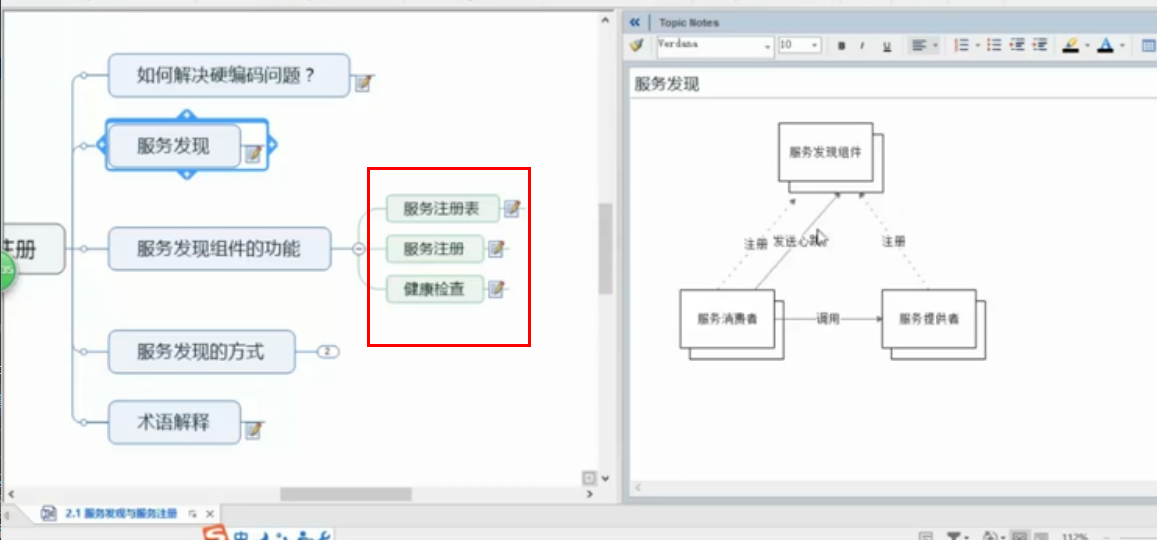


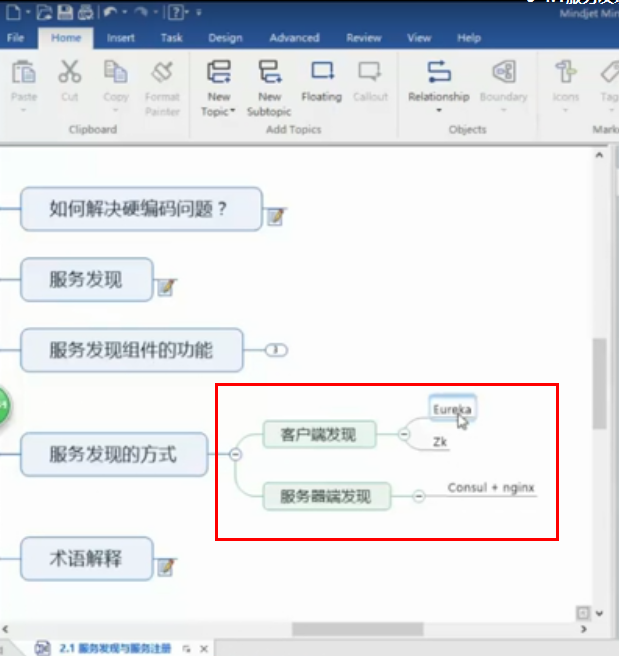
创建实体类



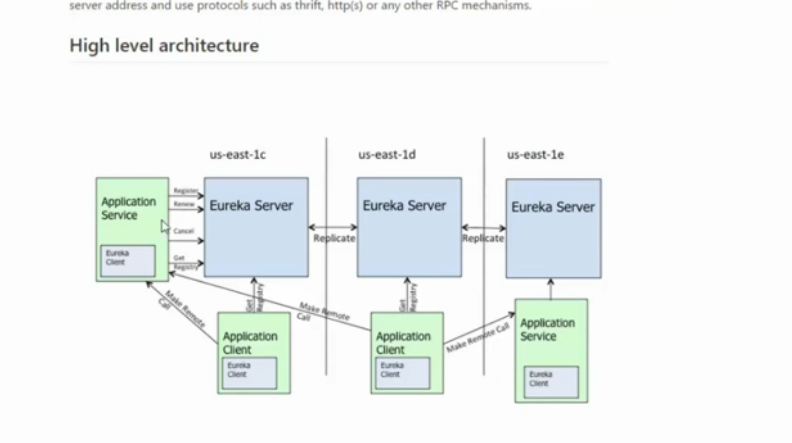
服务发现和服务注册



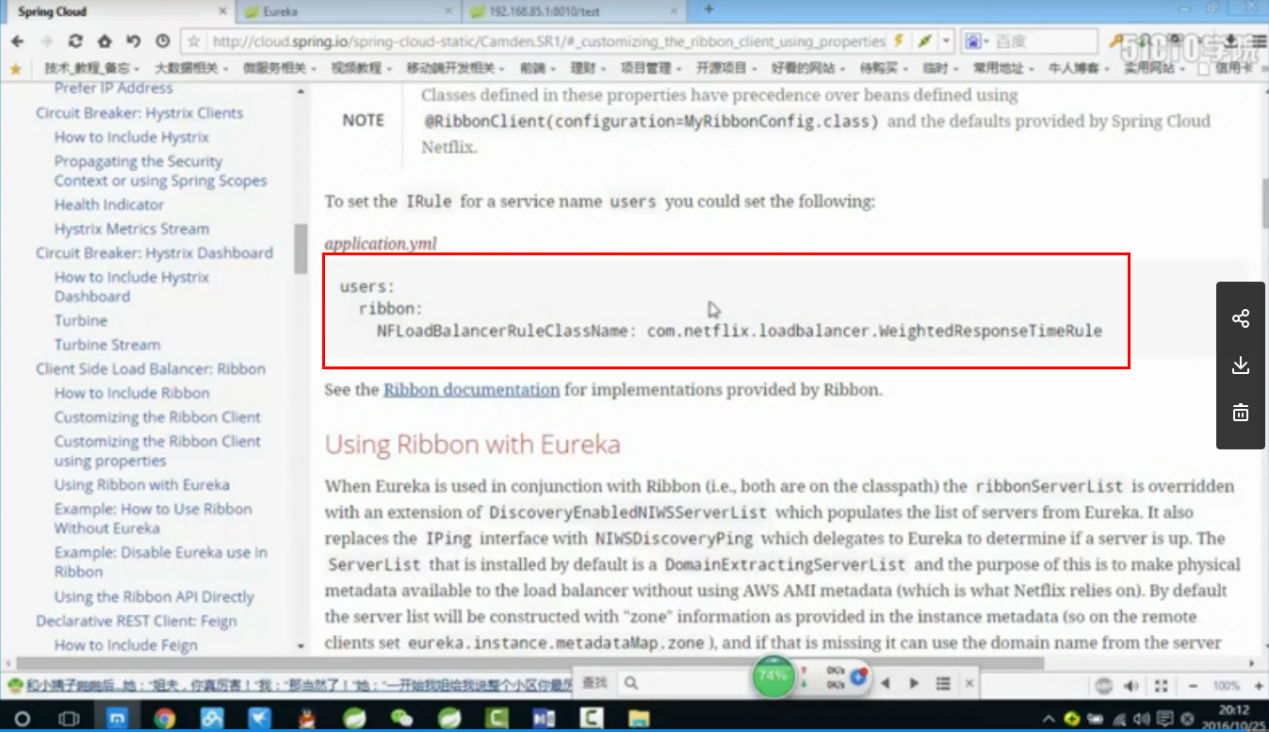




服务发现组件 Eureka



Application.yml文件中配置自定义的ribbon client



How to use Feign

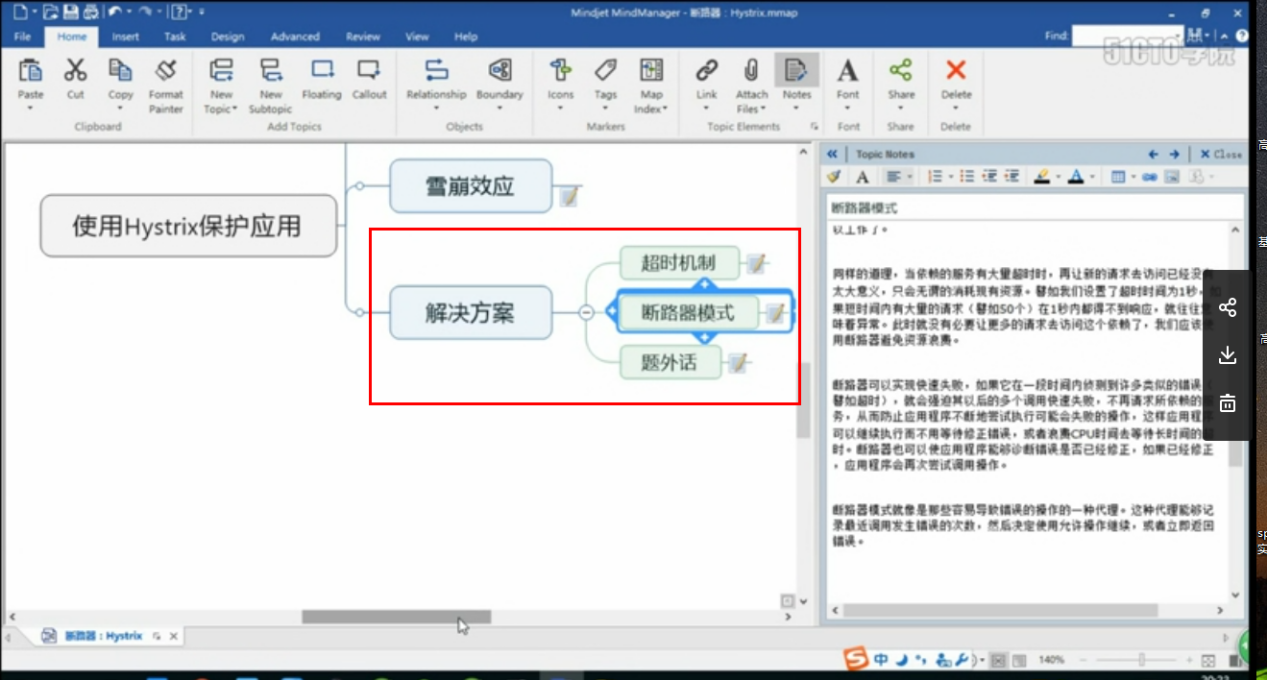
1. 引入jar包
2. 在启动类中加入@EnableFeignClients
3. 写feign接口

|  |
| --- |
| @FeignClient("microservice-provider-user")  public interface UserFeignClient {  @RequestMapping(value = "/simple/{id}", method = RequestMethod.GET)  public User findById(@PathVariable("id") Long id); // 两个坑：1. @GetMapping不支持 2. @PathVariable得设置value  @RequestMapping(value = "/user", method = RequestMethod.POST)  public User postUser(@RequestBody User user);  // 该请求不会成功，只要参数是复杂对象，即使指定了是GET方法，feign依然会以POST方法进行发送请求。可能是我没找到相应的注解或使用方法错误。  // 如勘误，请@lilizhou2008 eacdy0000@126.com  @RequestMapping(value = "/get-user", method = RequestMethod.GET)  public User getUser(User user);  } |

1. 在controller中引用

|  |
| --- |
| @RestController  public class MovieController {  @Autowired  private UserFeignClient userFeignClient;  @GetMapping("/movie/{id}")  public User findById(@PathVariable Long id) {  return this.userFeignClient.findById(id);  }  @GetMapping("/test")  public User testPost(User user) {  return this.userFeignClient.postUser(user);  }  @GetMapping("/test-get")  public User testGet(User user) {  return this.userFeignClient.getUser(user);  }  } |

Hystric 断路机制



第一步：引入jar包

第二步：在启动类上添加@EnableCircuitBreaker

第三步：在方法上面添加@HystrixCommand(fallbackMethod = "defaultStores")，然后再写一个fallback方法，注意：参数和返回值类型必须相同。

|  |
| --- |
| <dependency>  <groupId>org.springframework.cloud</groupId>  <artifactId>spring-cloud-starter-hystrix</artifactId>  </dependency> |
| @SpringBootApplication  @EnableEurekaClient  @EnableCircuitBreaker  public class MicroserviceSimpleConsumerMovieHystricApplication {  @Bean  public RestTemplate restTemplate() {  return new RestTemplate();  }  public static void main(String[] args) {  SpringApplication.run(MicroserviceSimpleConsumerMovieHystricApplication.class, args);  }  } |
| @RestController  public class MovieController {  @Autowired  private RestTemplate restTemplate;  @GetMapping("/movie/{id}")  @HystrixCommand(fallbackMethod = "findByIdFallBack")  public User findById(@PathVariable Long id) {  return this.restTemplate.getForObject("http://localhost:7900/simple/" + id, User.class);  }  public User findByIdFallBack(Long id) {  User user = new User();  user.setId(0L);  return user;  }  } |

在config配置文件中禁用单个feign

|  |
| --- |
| @Configuration  public class FooConfiguration {  @Bean  @Scope("prototype")  public Feign.Builder feignBuilder() {  return Feign.builder();  }  } |

Turbine:

1. 引入jar包
2. 在启动类上添加@EnableTurbine注解
3. 在application.yml文件中配置

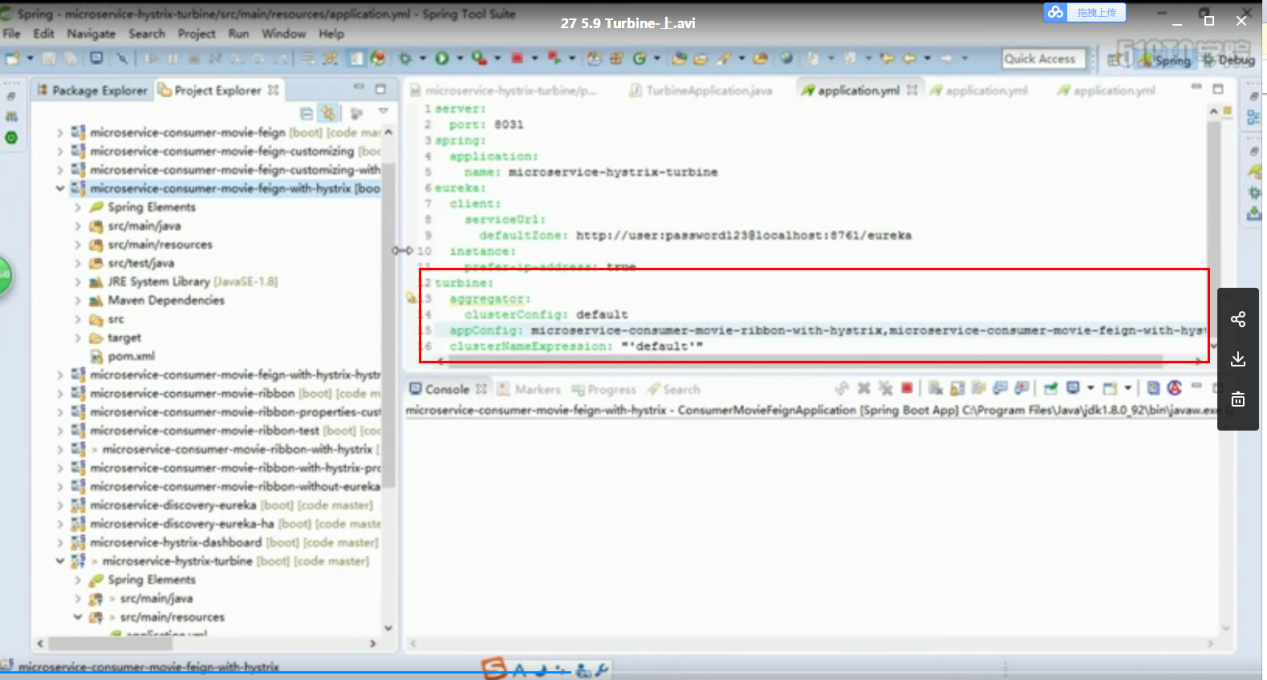
turbine:

aggregator:

clusterConfig: CUSTOMERS

appConfig: customers

监控多个服务时，在application.yml文件中配置



|  |
| --- |
| turbine:  appConfig: customers,stores  clusterNameExpression: "'default'" |

配置完毕后，启动服务，在浏览器中访问url地址：

http://my.turbine.server:8080/turbine.stream?cluster=CLUSTERNAME

在application.yml文件中配置log日志

logging:

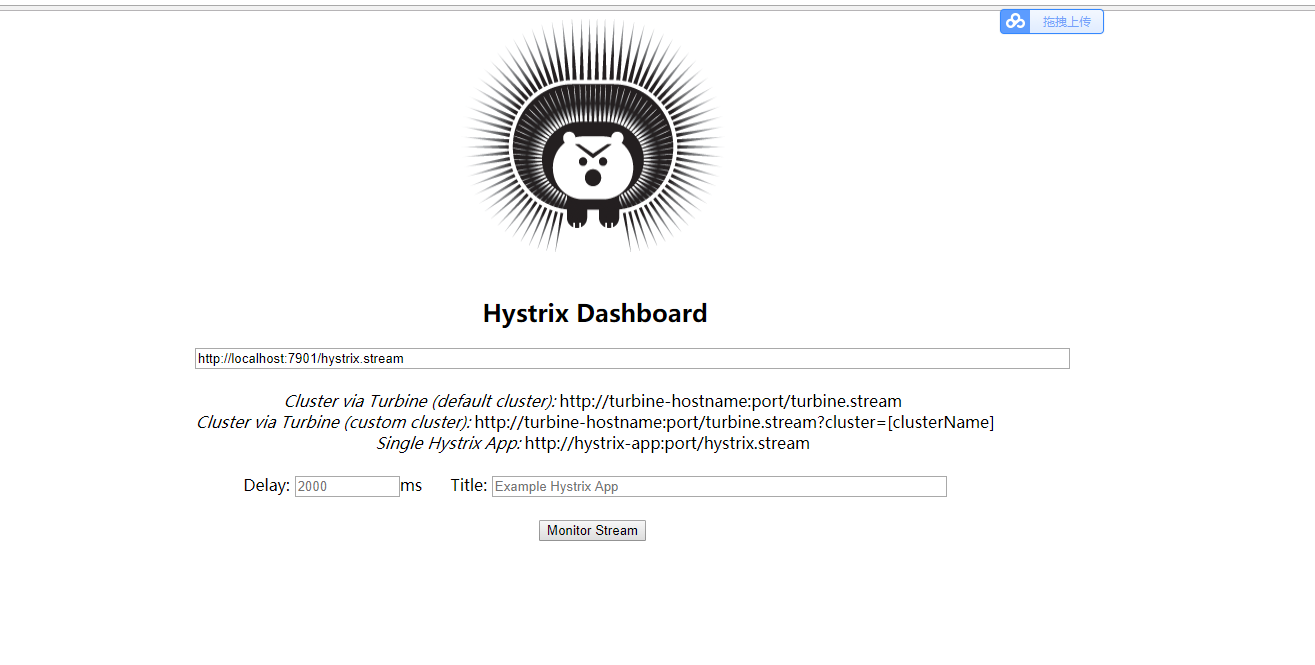
level:

root: INFO

com.netflix.turbine.monitor: DEBUG

## **Hystrix Dashboard**

启动服务，访问http://localhost:7901/hystrix.stream ,访问http://localhost:8030/hystrix 地址，会出现下面的页面。



APIGetway:服务网关

