skywalking-study

1.网址

```
1.官网
https://skywalking.apache.org/
2.文档
https://skywalking.apache.org/docs/main/latest/en/setup/backend/backend-setup/

3.github项目文档
https://github.com/fanda521/skywalking-project

4.下载指定的版本的skywalking
https://archive.apache.org/dist/skywalking/

agent 单独下载，新版本拆分开了
https://archive.apache.org/dist/skywalking/java-agent/8.8.0/


5.gateway 默认是不记录在里面的，需要把agent的optionPlugin 中的gateway的jar复制到agent中的plugins中才能看到记录，下面的是加了之后能看到的
```

2.注意点

1.项目jar

```
本身不用引入所谓的skywalking-spring-boot-starer,因为压根就没有这个，也不需要引入；除非定制化监控端点，需要引入下面这个jar
<dependency>
    <groupId>org.apache.skywalking</groupId>
    <artifactId>apm-toolkit-trace</artifactId>
</dependency>

```

2.启动需要探测的项目参数

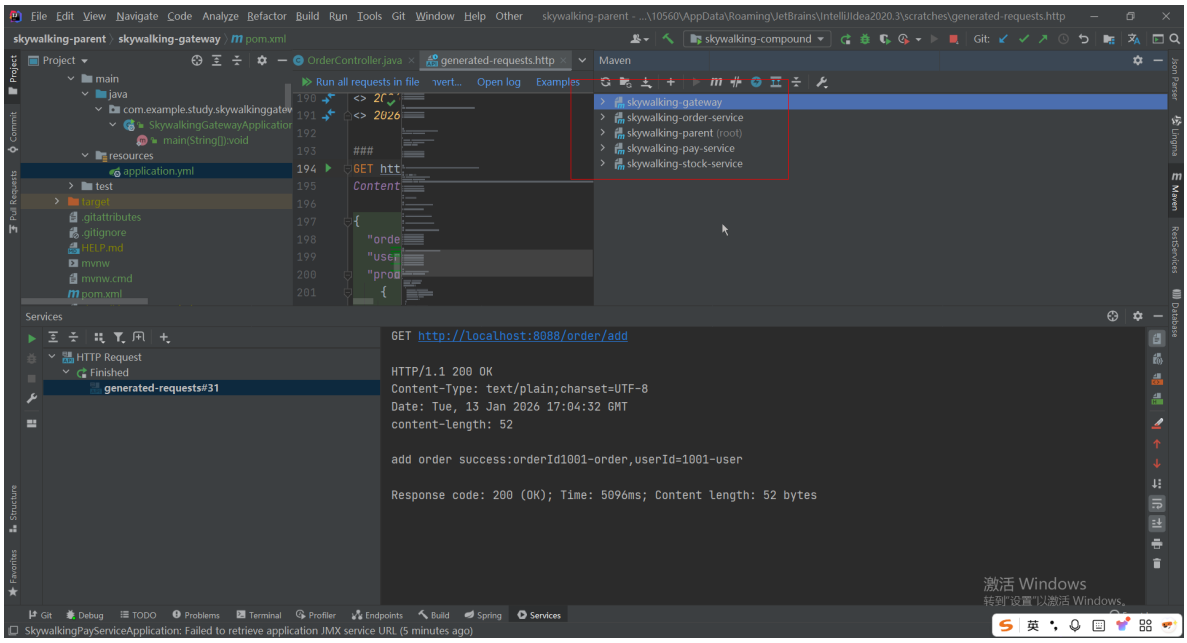
```
java -javaagent:/path/to/skywalking-agent/skywalking-agent.jar \
-Dskywalking.agent.service_name=your-spring-boot-app \
-Dskywalking.collector.backend_service=127.0.0.1:11800 \
-jar your-spring-boot-app.jar
```

3.项目版本坑

```
1.如果你的skywalking是8以上的需要jdk11,否则启动失败
```

3.项目结构

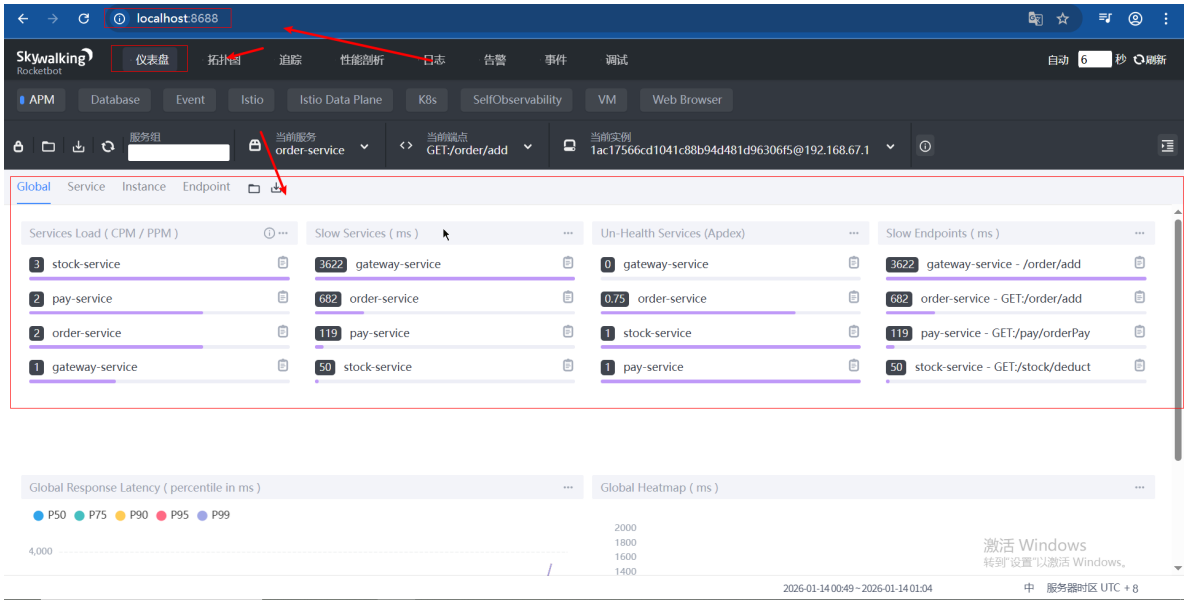
```
1.skywalking-parent
2.skywalking-gateway
3.skywalking-order-service
4.skywalking-pay-service
5.skywalking-stock-service
```

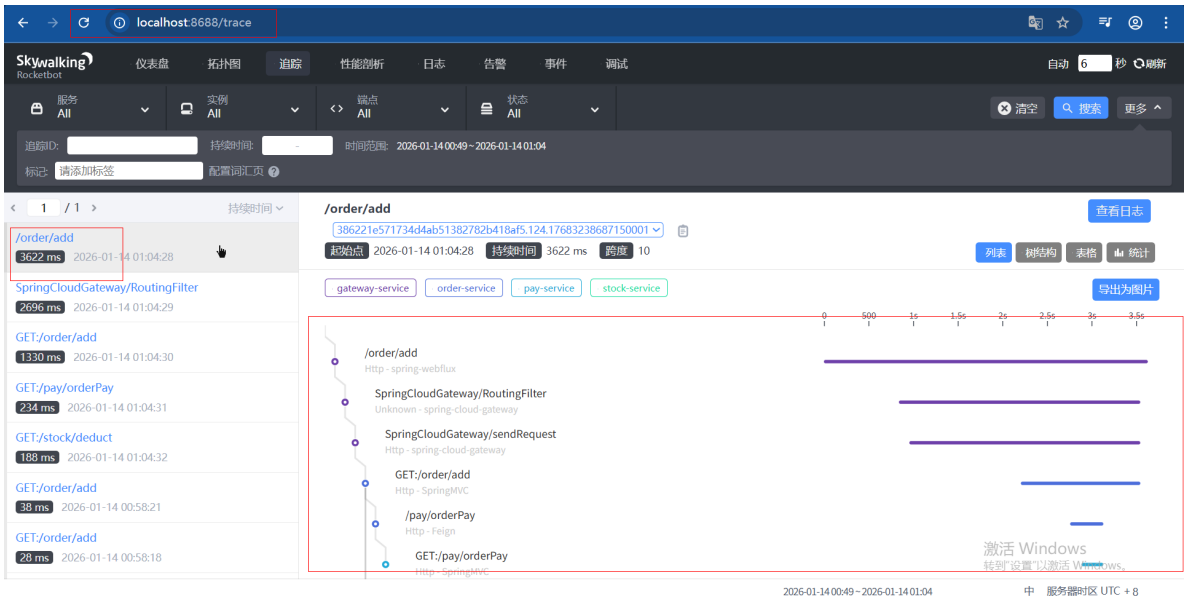


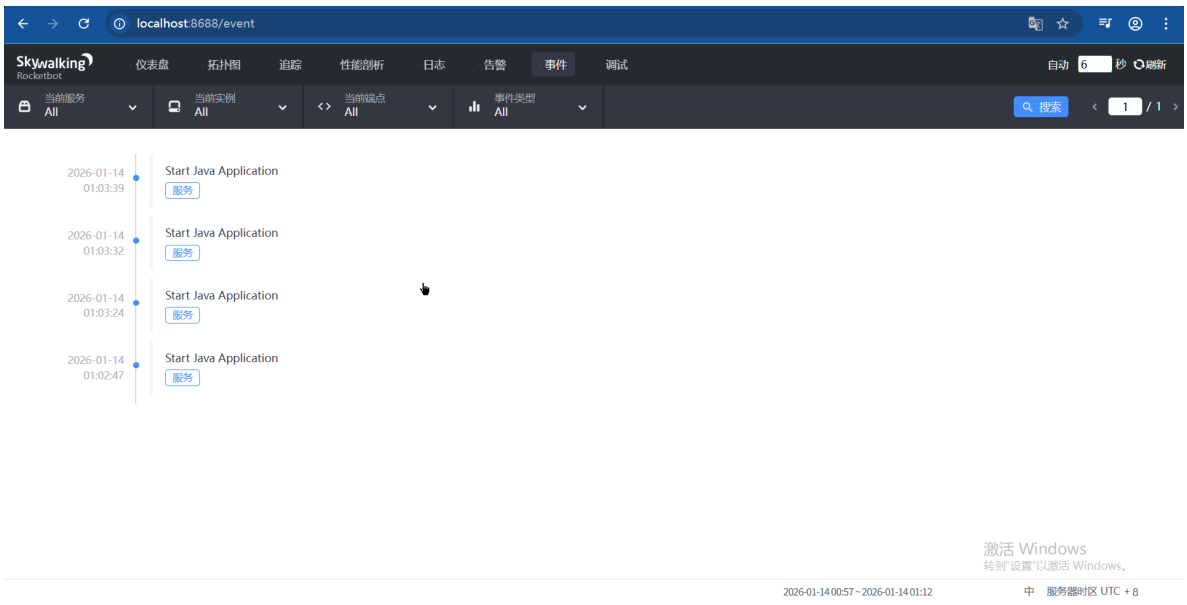
1.启动项目时的vm option

```
-javaagent:S:\Professions\skywalking\soft\apache-skywalking-java-agent-8.8.0\skywalking-agent/skywalking-agent.jar
-Dskywalking.agent.service_name=gateway-service
-Dskywalking.collector.backend_service=127.0.0.1:11800
```

2.skywalking-ui



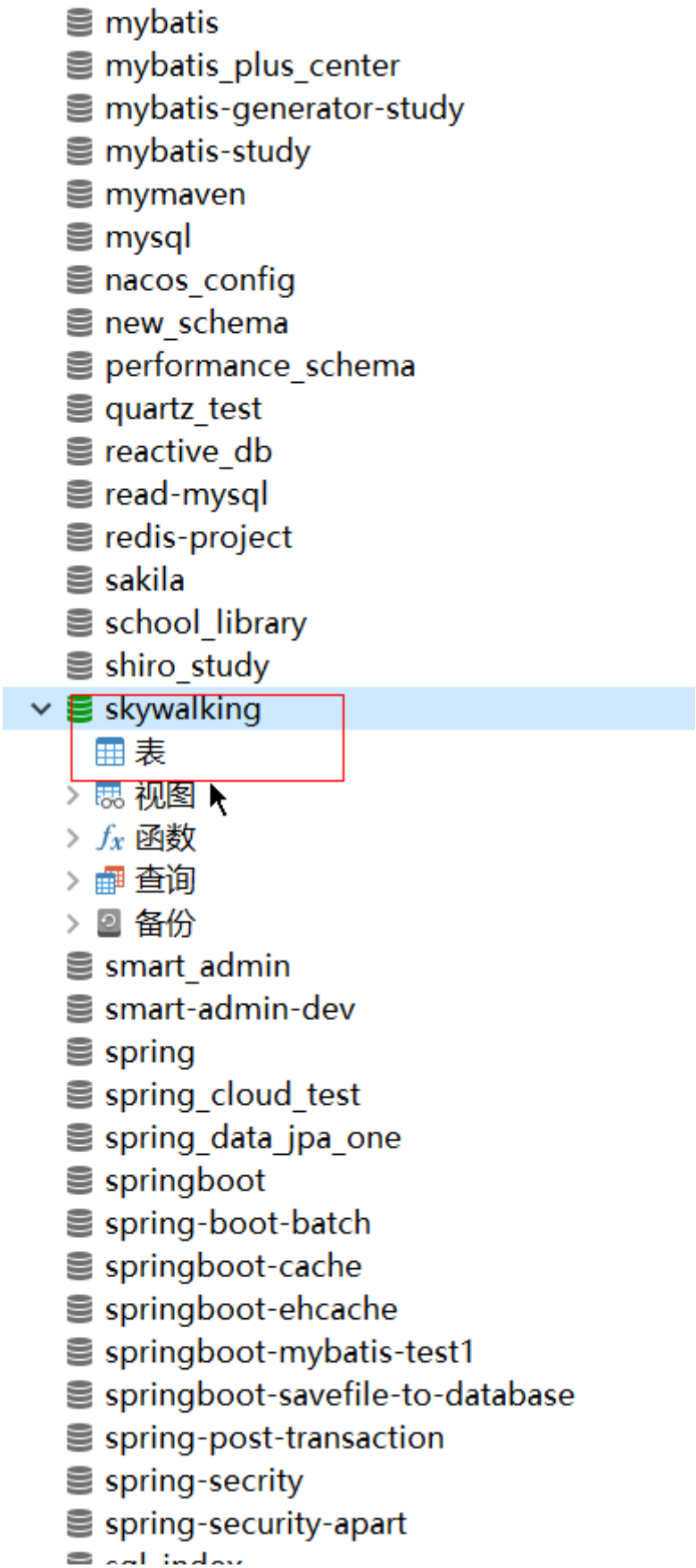


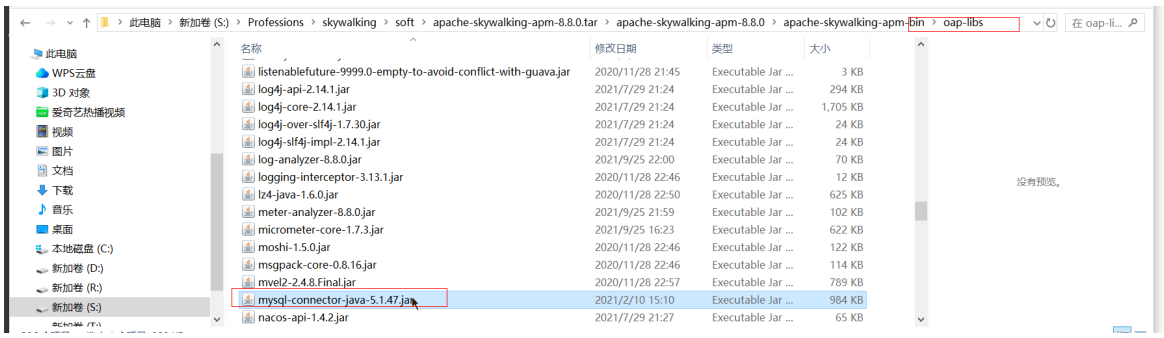


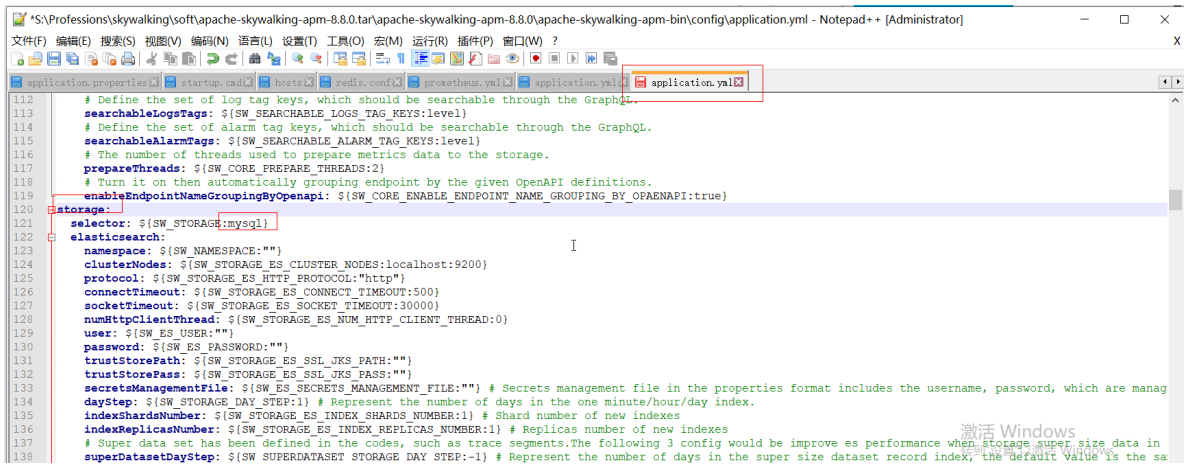
4.持久化放入mysql

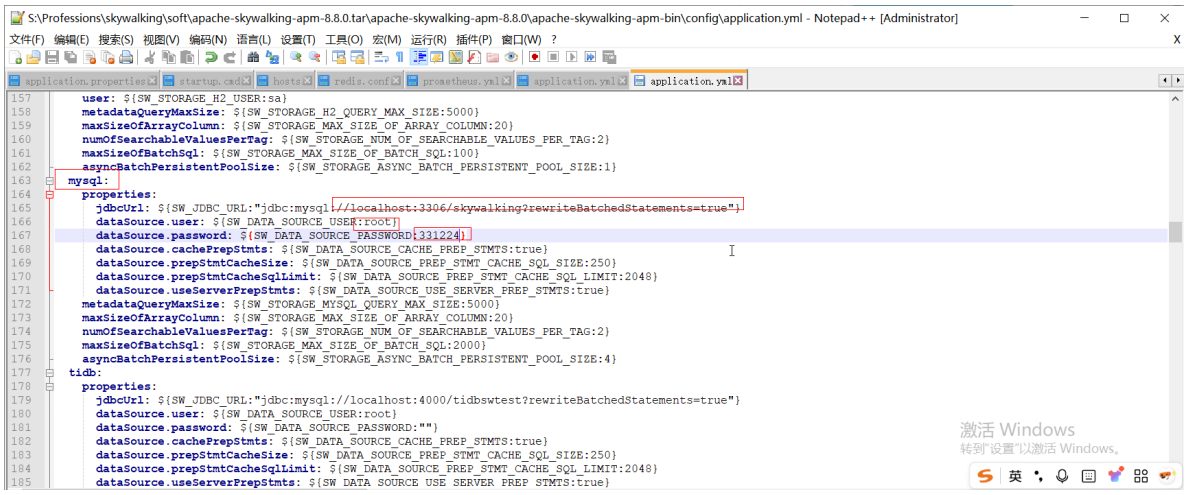
1.修改skywalking配置文件

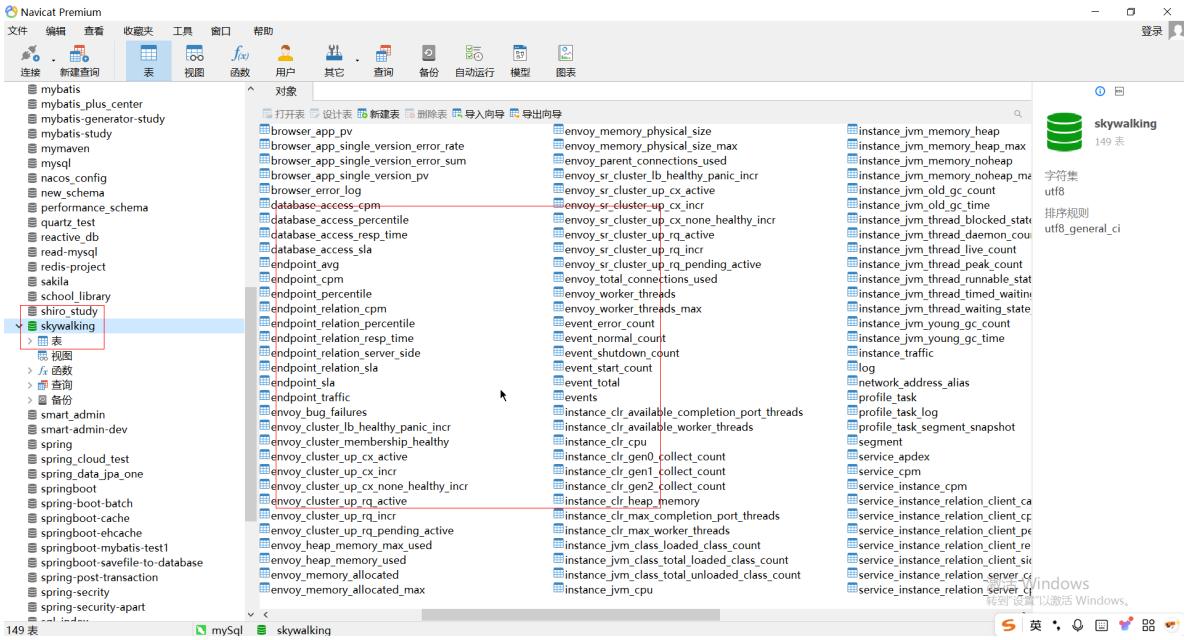
```
1.先建一个空的数据库skywalking
2.将mysql驱动的jar放入oap-libs中
3.修改配置
4.重启skywalking
```


mybatis
mybatis_plus_center
mybatis-generator-study
mybatis-study
mymaven
mysql
nacos_config
new_schema
performance_schema
quartz_test
reactive_db
read-mysql
redis-project
sakila
school_library
shiro_study
skywalking
表
视图
函数
查询
备份
smart_admin
smart-admin-dev
spring
spring_cloud_test
spring_data_jpa_one
springboot
spring-boot-batch
springboot-cache
springboot-ehcache
springboot-mybatis-test1
springboot-savefile-to-database
spring-post-transaction
spring-secrity
spring-security-apart


此电脑 › 新加卷 (S:) › Professions › skywalking › soft › apache-skywalking-apm-8.8.0.tar › apache-skywalking-apm-8.8.0 › apache-skywalking-apm-bin › oap-libs
名称
修改日期
类型
大小
listenablefuture-9999.0-empty-to-avoid-conflict-with-guava.jar 2020/11/28 21:45 Executable Jar ... 3 KB
log4j-api-2.14.1.jar 2021/7/29 21:24 Executable Jar ... 294 KB
log4j-core-2.14.1.jar 2021/7/29 21:24 Executable Jar ... 1,705 KB
log4j-over-slf4j-1.7.30.jar 2021/7/29 21:24 Executable Jar ... 24 KB
log4j-slf4j-impl-2.14.1.jar 2021/7/29 21:24 Executable Jar ... 24 KB
log-analyzer-8.8.0.jar 2021/9/25 22:00 Executable Jar ... 70 KB
logging-interceptor-3.13.1.jar 2020/11/28 22:46 Executable Jar ... 12 KB
lz4-java-1.6.0.jar 2020/11/28 22:50 Executable Jar ... 625 KB
meter-analyzer-8.8.0.jar 2021/9/25 21:59 Executable Jar ... 102 KB
micrometer-core-1.7.3.jar 2021/9/25 16:23 Executable Jar ... 622 KB
moshi-1.5.0.jar 2020/11/28 22:46 Executable Jar ... 122 KB
msgpack-core-0.8.16.jar 2020/11/28 22:46 Executable Jar ... 114 KB
mvel2-2.4.8.Final.jar 2020/11/28 22:57 Executable Jar ... 789 KB
mysql-connector-java-5.1.47.jar 2021/2/10 15:10 Executable Jar ... 984 KB
nacos-api-1.4.2.jar 2021/7/29 21:27 Executable Jar ... 65 KB
没有预览。


*S:\Professions\skywalking\soft\apache-skywalking-apm-8.8.0.tar\apache-skywalking-apm-8.8.0\apache-skywalking-apm-bin\config\application.yml - Notepad++ [Administrator]
application.yml
storage:
selector: ${SW_STORAGE:mysql}
elasticsearch:
激活 Windows


S:\Professions\skywalking\soft\apache-skywalking-apm-8.8.0.tar\apache-skywalking-apm-8.8.0\apache-skywalking-apm-bin\config\application.yml - Notepad++ [Administrator]
application.yml
mysql:
properties:
jdbcUrl: ${SW_JDBC_URL:"jdbc:mysql://localhost:3306/skywalking?rewriteBatchedStatements=true"}
dataSource.user: ${SW_DATA_SOURCE_USER:root}
dataSource.password: ${SW_DATA_SOURCE_PASSWORD:331224}
tidb:
properties:
jdbcUrl: ${SW_JDBC_URL:"jdbc:mysql://localhost:4000/tidbswtest?rewriteBatchedStatements=true"}
dataSource.user: ${SW_DATA_SOURCE_USER:root}
dataSource.password: ${SW_DATA_SOURCE_PASSWORD:""}
激活 Windows



5.自定义链路

1.注解

```
@Trace：标记该方法需要被 SkyWalking 追踪，@Tag 必须配合 @Trace 使用才生效，否则无法采集标签信息。
@Tag：单个标签定义，核心属性有两个：
key：标签名称（在 SkyWalking UI 中展示的标签键）。
value：标签值表达式，支持 SkyWalking 内置表达式，核心常用表达式：
args[index]：获取方法入参，index 为入参下标（从 0 开始），对应方法参数列表的顺序。
returnedObj：获取方法的返回结果（仅当方法正常返回时有效，异常时无法获取）。
@Tags：用于包裹多个 @Tag 注解，解决单个方法需要定义多个标签的场景（Java 不支持单个注解重复标注）。
```

2.jar

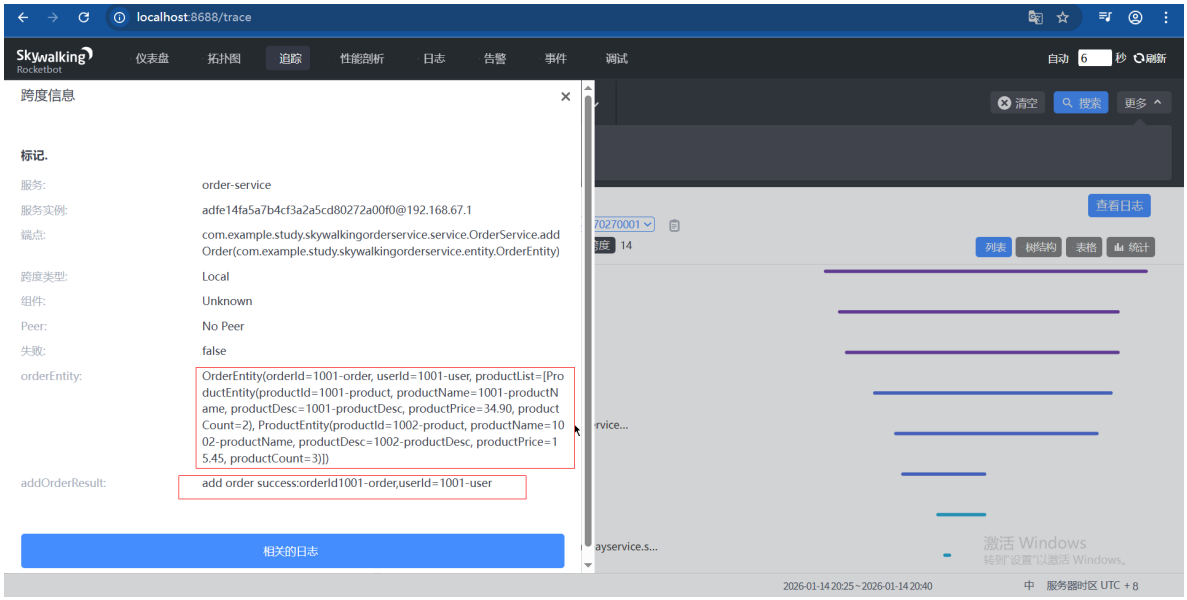
```xml
<!-- SkyWalking 追踪工具包，提供 @Trace/@Tag 等注解 -->
<dependency>
    <groupId>org.apache.skywalking</groupId>
    <artifactId>apm-toolkit-trace</artifactId>
    <version>8.16.0</version> <!-- 替换为你的 SkyWalking 版本 -->
    <scope>provided</scope>
</dependency>
```

3.实际使用

```
@Trace // 标记该方法被 SkyWalking 追踪，必须添加
    @Tags({
            // 采集第一个入参 orderId（下标 0）
            @Tag(key = "orderEntity", value = "arg[0]"),
            @Tag(key = "addOrderResult", value = "returnedObj")
    })
    public String addOrder(OrderEntity orderEntity) throws JsonProcessingException {
        // 1.创建定点
```

```
        log.info("创建订单成功:orderEntity={}", objectMapper.writeValueAsString(orderEntity));
        // 2.支付
        BigDecimal amt = BigDecimal.ZERO;
        List<ProductEntity> productList = orderEntity.getProductList();
        if (productList != null && productList.size() > 0) {
            for (ProductEntity productEntity : productList) {
                amt = amt.add(productEntity.getProductPrice().multiply(new BigDecimal(productEntity.getProductCount())));
            }
        }
        payOpenFeignClient.pay(orderEntity.getOrderId(), orderEntity.getUserId(),amt);
        // 3.扣除库存
        for (ProductEntity productEntity : orderEntity.getProductList()) {
            stockOpenFeignClient.deduct(productEntity.getProductId(), productEntity.getProductCount());
        }
        return "add order success:orderId" + orderEntity.getOrderId() + ",userId=" + orderEntity.getUserId();

    }
```

4.效果展示



6.日志

1.pom

```
<dependency>
            <groupId>org.apache.skywalking</groupId>
            <artifactId>apm-toolkit-logback-1.x</artifactId>
            <version>8.8.0</version>
        </dependency>
```

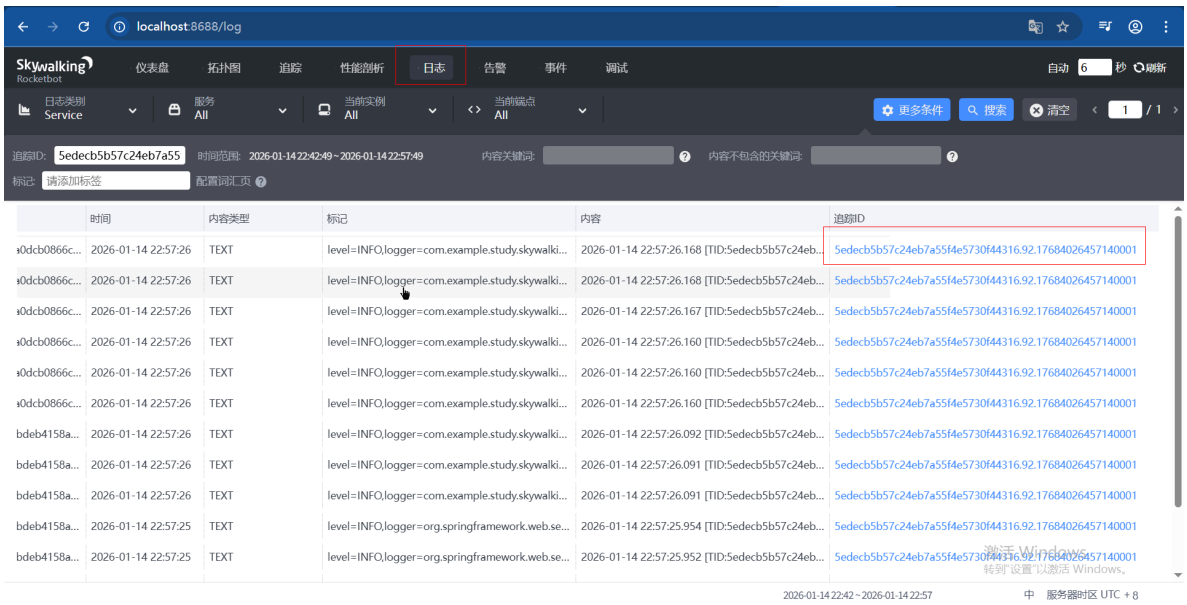
2.logback-srping.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration scan="true" scanPeriod="60 seconds">
    <!-- 1. 控制台输出 Appender（保留原有，方便本地调试） -->
    <appender name="CONSOLE" class="ch.qos.logback.core.ConsoleAppender">
        <encoder class="ch.qos.logback.core.encoder.LayoutWrappingEncoder">
            <!-- 日志格式：包含 TraceID（%tid）、时间、级别、类名、日志内容 -->
            <layout class="org.apache.skywalking.apm.toolkit.log.logback.v1.x.TraceIdPatternLogbackLayout">
                <pattern>%d{yyyy-MM-dd HH:mm:ss.SSS} [%tid] [%thread] %-5level %logger{36} - %msg%n</pattern>
            </layout>
        </encoder>
    </appender>

    <!-- 2. SkyWalking 日志上报 Appender（核心：将日志上报到 OAP） -->
    <appender name="SKYWALKING" class="org.apache.skywalking.apm.toolkit.log.logback.v1.x.log.GRPCLogClientAppender">
        <!-- 日志格式：必须包含 %tid，否则无法关联链路 -->
        <encoder class="ch.qos.logback.core.encoder.LayoutWrappingEncoder">
            <layout class="org.apache.skywalking.apm.toolkit.log.logback.v1.x.TraceIdPatternLogbackLayout">
                <pattern>%d{yyyy-MM-dd HH:mm:ss.SSS} [%tid] [%thread] %-5level %logger{36} - %msg%n</pattern>
            </layout>
        </encoder>
    </appender>

    <!-- 3. 根日志配置：同时输出到控制台和 SkyWalking -->
    <root level="INFO">
        <appender-ref ref="CONSOLE"/>
        <appender-ref ref="SKYWALKING"/>
    </root>
</configuration>
```

3.本地控制台

```
2026-01-14 22:57:25.716 [TID:5edecb5b57c24eb7a55f4e5730f44316.92.17684026457140001] [http-nio-8084-exec-5] INFO  c.e.s.s.service.OrderService - 创建订单成功:orderEntity={"orderId":"1001-order","userId":"1001-user","productList":[{"productId":"1001-product","productName":"1001-productName","productDesc":"1001-productDesc","productPrice":34.90,"productCount":2},{"productId":"1002-product","productName":"1002-productName","productDesc":"1002-productDesc","productPrice":15.45,"productCount":3}]}

TID:5edecb5b57c24eb7a55f4e5730f44316.92.17684026457140001
```

4.skywalking-ui->日志



7.告警

1.yaml

```
rules同层级

webhooks:
#  - http://127.0.0.1/notify/
#  - http://127.0.0.1/go-wechat/
  - http://127.0.0.1:8084/skywalking/alarm/receive

```

2.代码

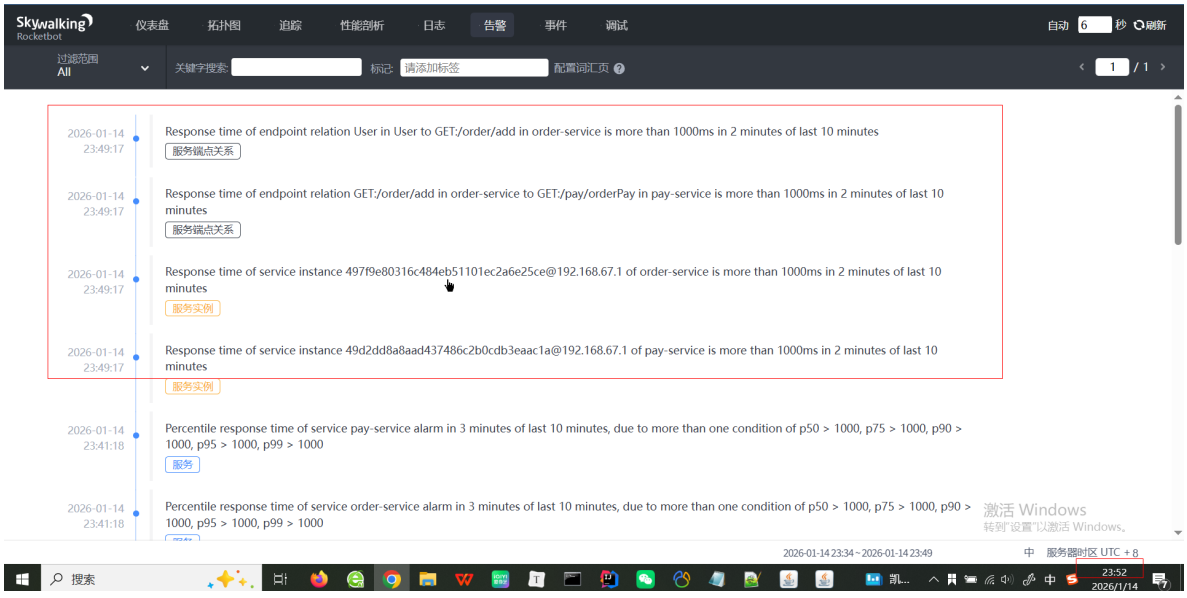
```
pay-service
加睡眠2秒，然后不停地掉接口
```

3.配置工告警调用的接口

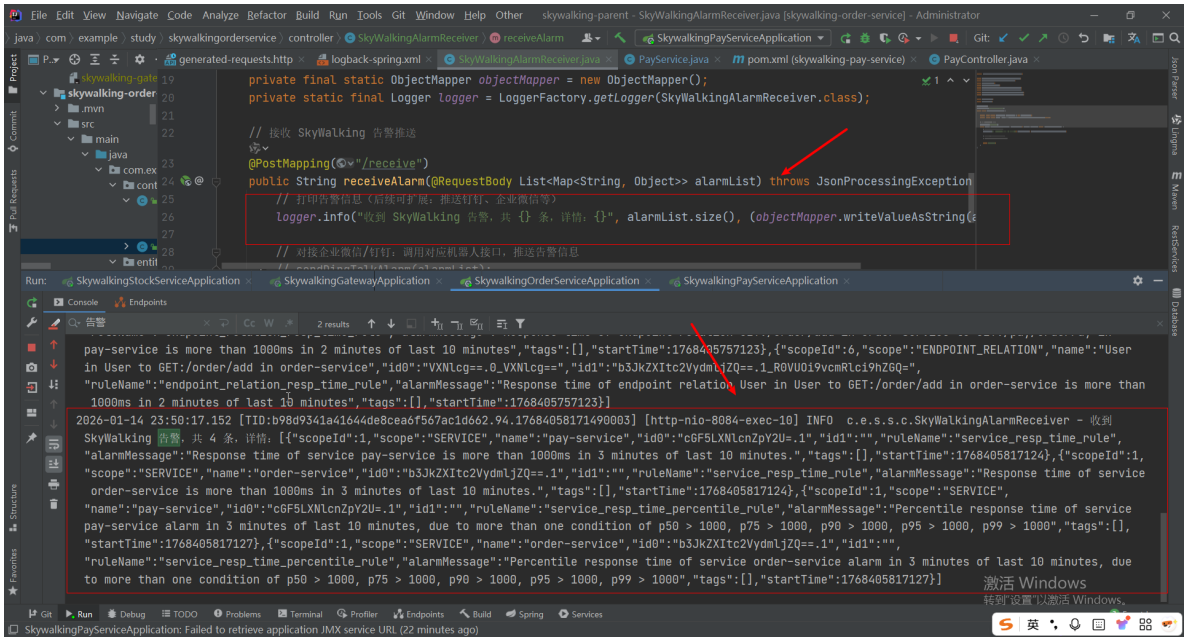
```


package com.example.study.skywalkingorderservice.controller;

import com.fasterxml.jackson.core.JsonProcessingException;
import com.fasterxml.jackson.databind.ObjectMapper;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import java.util.List;
```

```java
import java.util.Map;

@RestController
@RequestMapping("/skywalking/alarm")
public class SkyWalkingAlarmReceiver {

    private final static ObjectMapper objectMapper = new ObjectMapper();
    private static final Logger logger =
LoggerFactory.getLogger(SkyWalkingAlarmReceiver.class);

    // 接收 SkyWalking 告警推送
    @PostMapping("/receive")
    public String receiveAlarm(@RequestBody List<Map<String, Object>>
alarmList) throws JsonProcessingException {
        // 打印告警信息（后续可扩展：推送钉钉、企业微信等）
        logger.info("收到 SkyWalking 告警，共 {} 条，详情：{}",
alarmList.size(), (objectMapper.writeValueAsString(alarmList)));

        // 对接企业微信/钉钉：调用对应机器人接口，推送告警信息
        // sendDingTalkAlarm(alarmList);

        return "success";
    }
}
```

4.观察skywalking-ui告警模块



5.日志




```
2026-01-14 23:50:17.152 [TID:b98d9341a41644de8cea6f567ac1d662.94.17684058171490003] [http-nio-8084-exec-10] INFO  c.e.s.s.c.SkyWalkingAlarmReceiver - 收到 SkyWalking 告警，共 4 条，详情：[{"scopeId":1,"scope":"SERVICE","name":"pay-service","id0":"cGF5LXNlcnZpY2U=.1","id1":"","ruleName":"service_resp_time_rule","alarmMessage":"Response time of service pay-service is more than 1000ms in 3 minutes of last 10 minutes.","tags":[],"startTime":1768405817124},{"scopeId":1,"scope":"SERVICE","name":"order-service","id0":"b3JkZXItc2VydmljZQ==.1","id1":"","ruleName":"service_resp_time_rule","alarmMessage":"Response time of service order-service is more than 1000ms in 3 minutes of last 10 minutes.","tags":[],"startTime":1768405817124},{"scopeId":1,"scope":"SERVICE","name":"pay-service","id0":"cGF5LXNlcnZpY2U=.1","id1":"","ruleName":"service_resp_time_percentile_rule","alarmMessage":"Percentile response time of service pay-service alarm in 3 minutes of last 10 minutes, due to more than one condition of p50 > 1000, p75 > 1000, p90 > 1000, p95 > 1000, p99 > 1000","tags":[],"startTime":1768405817127},{"scopeId":1,"scope":"SERVICE","name":"order-service","id0":"b3JkZXItc2VydmljZQ==.1","id1":"","ruleName":"service_resp_time_percentile_rule","alarmMessage":"Percentile response time of service order-service alarm in 3 minutes of last 10 minutes, due to more than one condition of p50 > 1000, p75 > 1000, p90 > 1000, p95 > 1000, p99 > 1000","tags":[],"startTime":1768405817127}]
```