springboot|springboot 集成 CAT 做系统监控

作者：xiaodaojava
原文链接：https://ld246.com/article/1574674549156
来源网站：链滴

javaDEMO

本网站记录了最全的各种JavaDEMO ,保证下载,复制就是可用的,包括基础的, 集合的, spring的, Mybats的等等各种,助力你从菜鸟到大牛,记得收藏哦~~

https://www.javastudy.cloud

CAT总述

官方参考地址:

https://github.com/dianping/cat

CAT是大众点评开源一款监控组件,这款监控以整体监控为主,即不能对每个请求,远程调用都做详细记录只保留了错误的,长调用等典型的调用以供开发人员查看

CAT中大致就两大部分, 一个是服务端,需要连接数据库, 一个就是发送数据的客户端

CAT服务端搭建

服务端我们依然选用Docker来搭建,先把cat整个项目下载从官网下载下来,地址:https://codeload.github.com/dianping/cat/zip/master

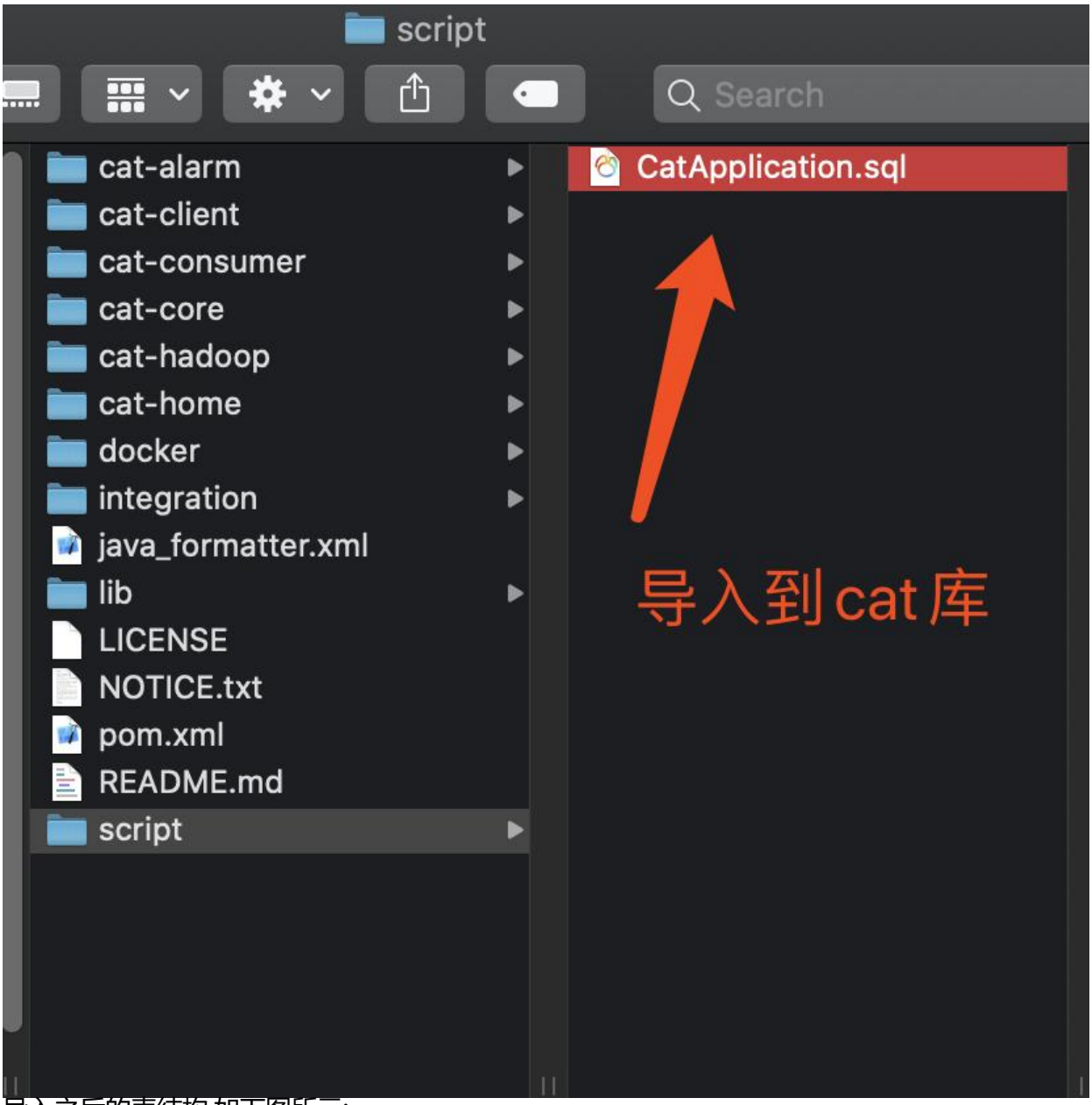
数据库配置

在数据库中创建一个名为cat的数据库

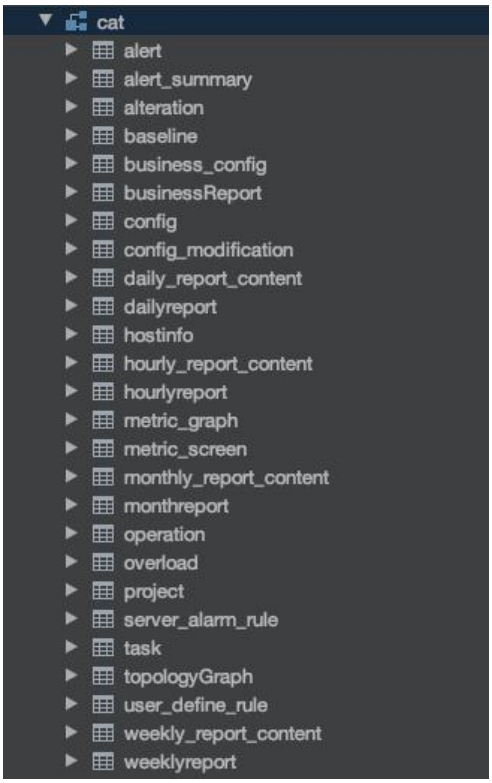
可使用sql语句,也可以用可视化界面,注意编码要选 utf8mb4

```
create schema cat default character set = utf8mb4;
```

然后把script/CatApplication.sql导入到数据库中



导入之后的表结构,如下图所示:



修改docker配置文件

然后我们需要修改以下配置文件

1. client.xml

修改ip为自己电脑的ip

```
<?xml version="1.0" encoding="utf-8"?>
<config mode="client">
   <servers>
      <server ip="192.168.40.107" port="2280" http-port="8080"/>
   </servers>
</config>
```

2. datasources.xml

这里要注意!!!! 一定要使用mysql 5.7的版本!!!!!

中间的url, user, password , 换成自己数据库的地址,用户名和密码

```
<properties>
        <driver>com.mysql.jdbc.Driver</driver>
        <url><![CDATA[jdbc:mysql://192.168.40.107:33309/cat]]></url>
        <user>root</user>
        <password>javastudy</password>
        <connectionProperties><![CDATA[useUnicode=true&characterEncoding=UTF-8&aut
Reconnect=true&socketTimeout=120000]]></connectionProperties>
    </properties>
```

3. Dockerfile

50行下面添加

ADD docker/client.xml /data/appdatas/cat/client.xml

把第51行的 ADD docker/datasources.sh /datasources.sh 删掉,因为这个是搜索jdbc-url,user,passd 把这些改成用环境变量来传递,我们上面改过datasources.xml之后,就不用这个了

最后两行, 改成:

```
# 端口加一个 2280
EXPOSE 8080 2280
# 执行tomcat的启动命令
CMD catalina.sh run
```

从45行开始的Dockerfile 如下:

```
44
45  COPY . /root/workspace/cat
46  WORKDIR /root/workspace/cat
47
48  RUN set -ex && mvn clean install -DskipTests
49  RUN cp cat-home/target/*.war $CATALINA_HOME/webapps/cat.war
50  ADD docker/datasources.xml /data/appdatas/cat/datasources.xml
51  ADD docker/client.xml /data/appdatas/cat/client.xml
52  RUN sed -i "s/port=\"8080\"/port=\"8080\"\ URIEncoding=\"utf-8\"/g" $CATALINA_HOME/conf/server.xml
53
54  EXPOSE 8080 2280
55  CMD catalina.sh run
```

开始docker镜像文件构建

这里要注意了, 要在cat-master目录执行这个命令

```
# 在dockerfile 的上级目录,也就是cat-master这个目录执行这个命令
# 用 -f 指定dockerFile的路径
docker build -f  docker/Dockerfile   -t cat:0.1 .
```

因为是从源码来编译,下载依赖,所以过程会有些慢

最后可看到已成功打好镜像

```
Removing intermediate container 5731d060e477
 ---> 72c882c840a5
Successfully built 72c882c840a5
Successfully tagged cat:0.1
```

运行镜像

```
# 需要绑定两个端口,8080 和 2280
docker run -d --name cat-server -p 8080:8080 -p 2280:2280 cat:0.1
```

浏览器输入地址: http://192.168.40.107:8080/cat/r 可看到CAT界面

CAT客户端

CAT 客户端只有几以下几个部分组成

1. 引入CAT的依赖
2. 针对要监控的不同模块,做不同的集成方案

可集成的模块可参考以下文档:

https://github.com/dianping/cat/tree/master/integration , 这里我们以springboot为例

添加cat依赖

这里要注意,cat-client并没有发布到中央仓库,有两种方法:

一.把cat-client.jar下载下来,然后上传到自己的私服

二.配置unidal的仓库地址:

```
repositories {

    mavenCentral()
    //重点在这里,添加unidal的地址
    maven {
        url 'http://unidal.org/nexus/content/repositories/releases/'
    }
}
```

在原有web的基础上Cat只用添加这一个依赖就可以了

```
compile group: 'com.dianping.cat', name: 'cat-client', version: '3.0.0'
```

完整依赖如图:

```
dependencies {
    //  Cat的依赖只用引这一个就可以了
    compile group: 'com.dianping.cat', name: 'cat-client', version: '3.0.0'


    implementation 'org.springframework.boot:spring-boot-starter-thymeleaf'
    implementation 'org.springframework.boot:spring-boot-starter-web'
    testImplementation('org.springframework.boot:spring-boot-starter-test') {
        exclude group: 'org.junit.vintage', module: 'junit-vintage-engine'
    }
}
```

添加测试的Controller代码

因为cat已启用8080端口,所以这里把程序端口改成了9090

```
server.port=9090
```

测试的Controller代码

```
@RestController
public class TestController {

    @GetMapping("testCat")
    public String testCat(){
        return "in testCat";
    }
}
```

整合进Cat的代码

Cat与springboot,web的集成以过滤器的形式,我们在Main函数所在类中中添加:

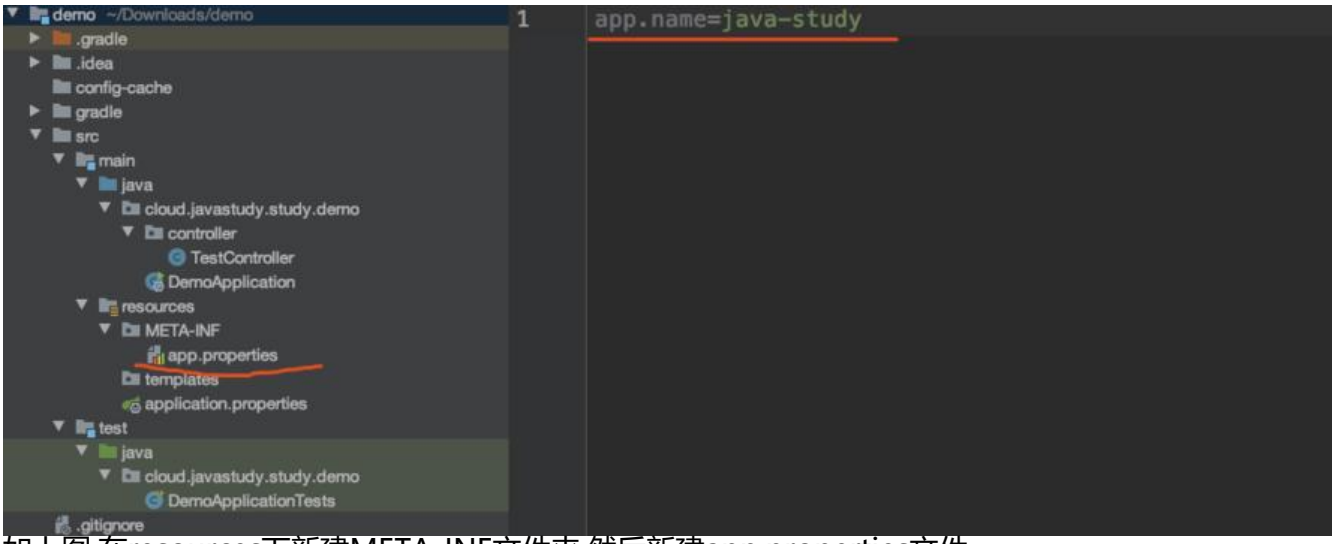
```
@Bean
    public FilterRegistrationBean catFilter() {

        FilterRegistrationBean registration = new FilterRegistrationBean();
        CatFilter filter = new CatFilter();
        registration.setFilter(filter);
        registration.addUrlPatterns("/*");
        registration.setName("cat-filter");
        registration.setOrder(1);
        return registration;
    }
```

在客户端配置cat服务端信息

现在要告诉客户端,服务端在哪里, 信息要往哪里发送

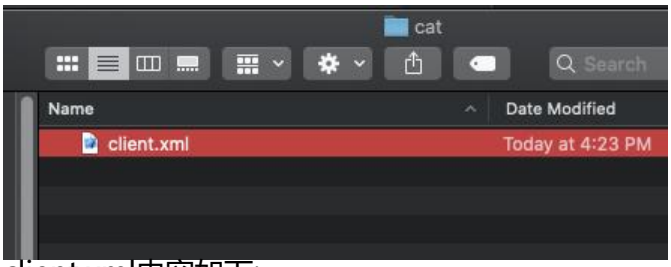
1.新建 app.properties



如上图,在resources下新建META-INF文件夹,然后新建app.properties文件

```
#  向cat 指定自己应用的名称
app.name=java-study
```

2.在电脑中建一个cat文件夹,把源码中的docker/client.xml 拷贝到cat文件夹下

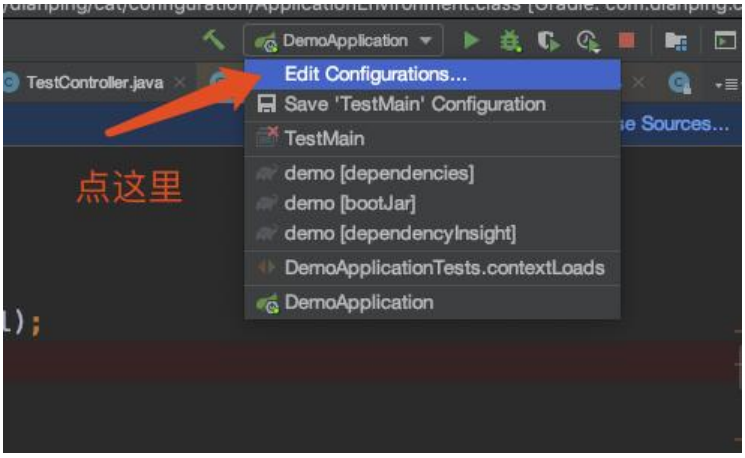


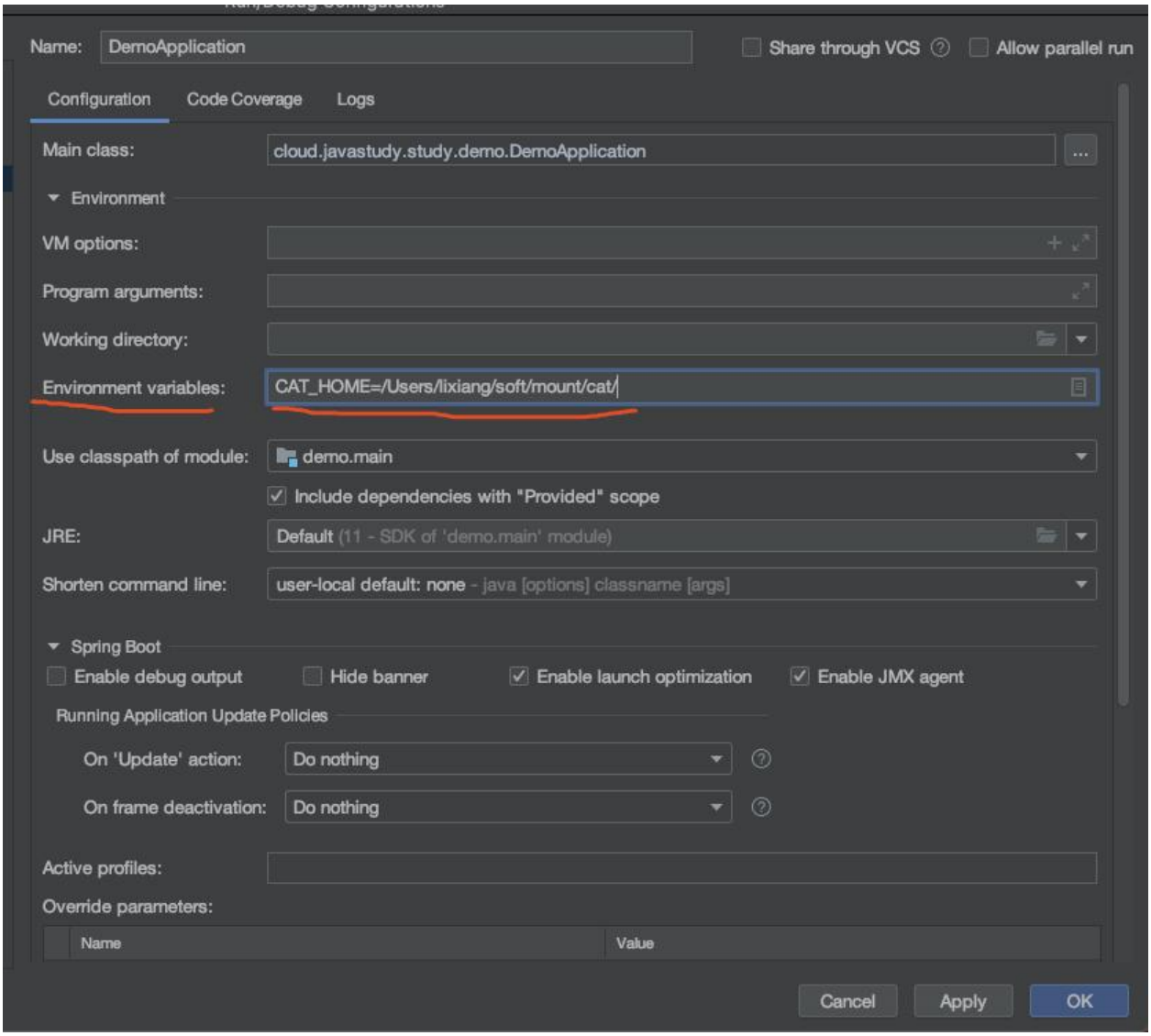
client.xml内容如下:

指定cat服务器的相关信息

```
<?xml version="1.0" encoding="utf-8"?>
<config mode="client">
    <servers>
        <server ip="192.168.40.107" port="2280" http-port="8080"/>
    </servers>
</config>
```

3. 配置运行项





在idea右上角运行配置中,设置CAT_HOME变量,指向我们刚才建cat目录的位置

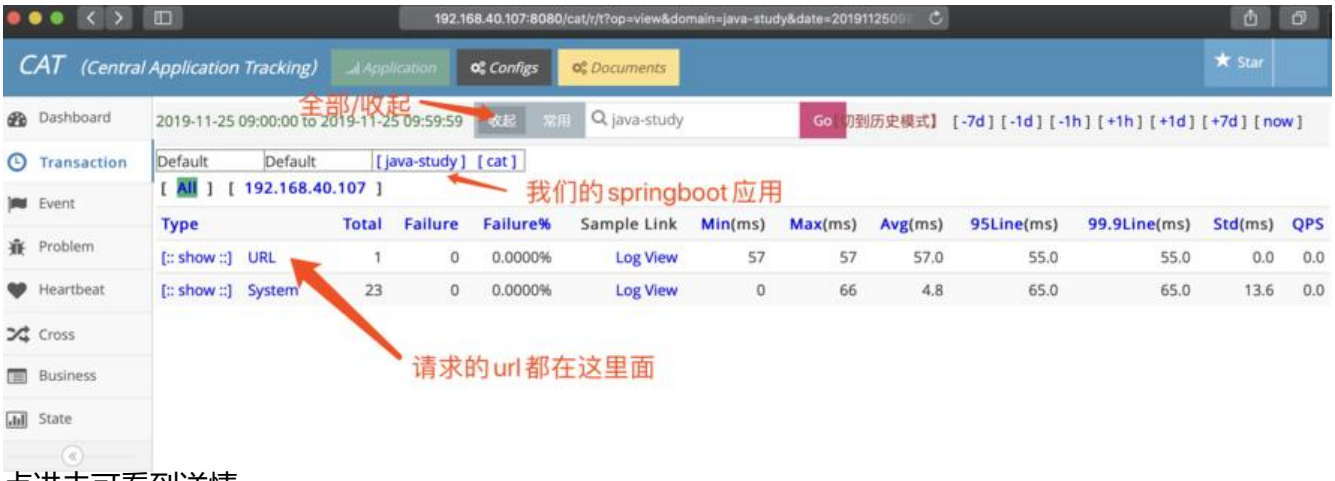
然后运行

结果测试

在浏览器输入http://localhost:9090/testCat能看到相应的返回结果.

in testCat

在Cat面板中



点进去可看到详情

DEMO总评

CAT的官方文档比较少,深入集成的坑比较多,但是CAT功能又是非常之强大,要注意CAT_HOME的坑,注意Mysql版本的坑, 等等, 同样, 本DEMO只用于学习测试, 在生产环境,还需要集群式部署cat,加油吧,年!!