1. cat的部署
   1. 准备条件

Linux 2.6以及之上（2.6内核才可以支持epoll），Mac以及Windows环境可以作为开发环境

Java 6，7，8

Maven 3.2.3+

MySQL 5.6

* 1. 打包

从github上下载整个项目，先在cat主目录里运行mvn clean install -DskipTests，再进入cat-home目录运行mvn package得到打包好的war包，改名为cat，放到tomcat目录下的webapps里

* 1. 配置

Linux\Mac 需要对/data/appdatas/cat和/data/applogs/cat有读写权限

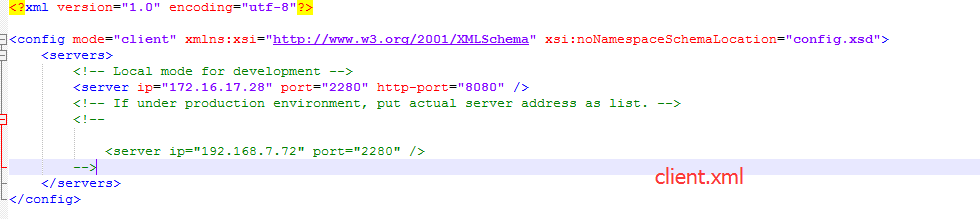
Windows 则是对系统运行盘下的/data/appdatas/cat和/data/applogs/cat有读写权限,如果cat服务运行在e盘的tomcat中，则需要对e:/data/appdatas/cat和e:/data/applogs/cat有读写权限。

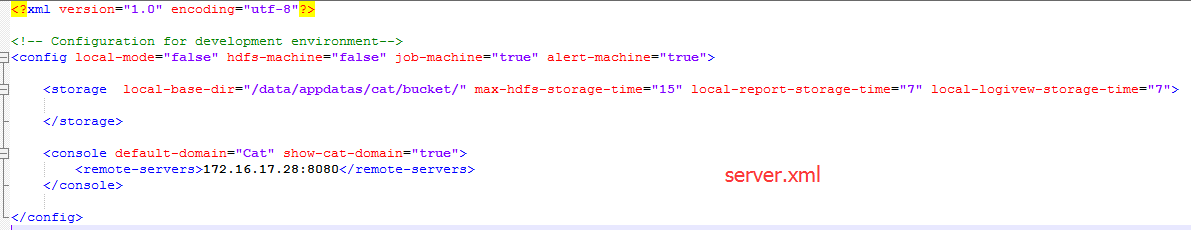
/data/appdatas/cat/ 下面需要的几个配置文件，配置模板文件在源码script 。script里有生成数据库的sql，有时需手动执行。

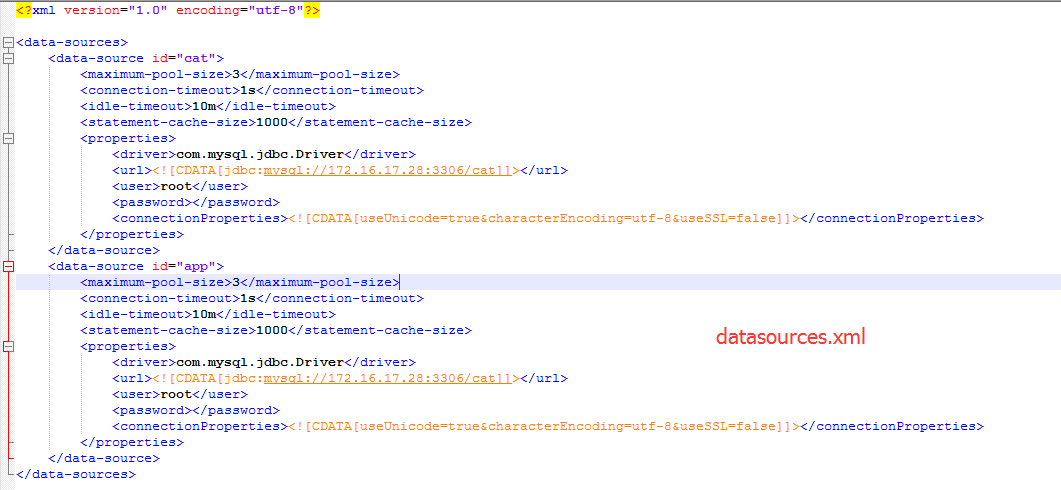
以下为例子。

具体详情访问：

http://unidal.org/cat/r/home?op=view&docName=deploy





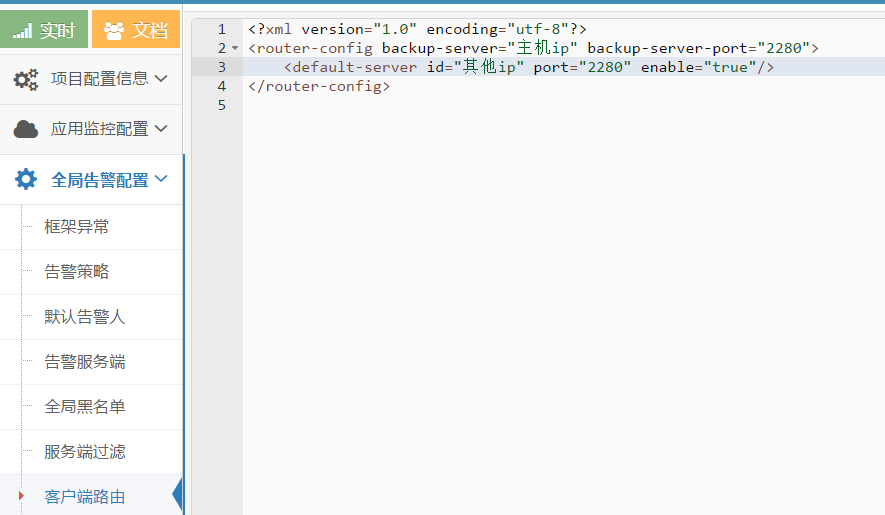


注意：如果多台服务器要实现共联，使用同样的配置文件就可以了。



server.xml配置里只需要一台服务器为true就可以了，其他的可都为false。

多台服务器还需要在网站里配置客户端路由。例子如下：

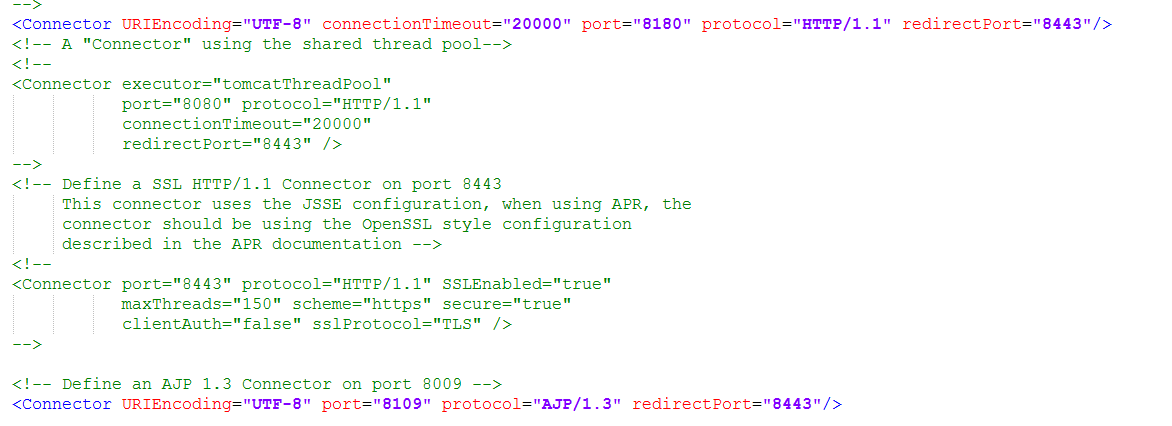


* 1. 运行

启动tomcat，访问 <http://localhost:8080/cat/r> 即可。配置界面的默认账号密码为catadmin/catadmin

1. 常见问题
   1. 打点日志乱码

修改tomcat的server.xml文件，通过添加编码方式解决。



在两个connector标签均添加属性URIEncoding="UTF-8"

* 1. 网页访问异常，请求异常

可能是cat启动netty出错，可能端口冲突。启动tomcat时查看\data\applogs\cat里的日志。例子如下。

* 1. 埋点代码执行成功却没显示

cat的埋点采用的的是异步处理。在埋点结束后的几秒后才上传。若执行埋点代码后程序就结束，可能上传不成功。

解决方法：在程序最后加入 sleep(5000); 试试。

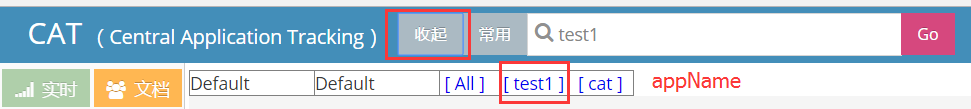
* 1. 其他帮助文档参考

https://github.com/dianping/cat/blob/master/框架埋点方案集成/Cat技术入门总结-0.1.0.doc

1. cat的埋点
   1. 客户端准备

在需要被监听的项目src/main/resources/META-INF目录下添加配置文件app.properties。app的内容为 ： **app.name**=**test1。**为项目名称。

app.name对应下图。在埋点成功后才会显示app name。



注意：我的app.properties是放在classes输出目录里bin/META-INF里的。具体项目放在的位置可能不同。可在埋点时进行调试。具体在com.dianping.cat.configuration.DefaultClientConfigManager 里的loadProjectName()方法。查看appName是否赋值成功。



* 1. 依赖包

maven项目里加入如下依赖。

<**dependency**>  
 <**groupId**>com.dianping.cat</**groupId**>  
 <**artifactId**>cat-client</**artifactId**>  
 <**version**>RELEASE</**version**>  
</**dependency**>

java项目里需要以下jar包。



* 1. Transaction埋点

Transaction t = Cat.*getProducer*().newTransaction(

CatConstants.TYPE\_SQL, **"this is transaction name "**);

yourBussiss();//处理业务

t.setStatus(Transaction.***SUCCESS***);  
t.complete();

newTransaction第一个参数为type，cat有提供一些type，用于判断transaction的业务类型，如上的可以告诉cat这是一个sql业务的。type也可自定义。

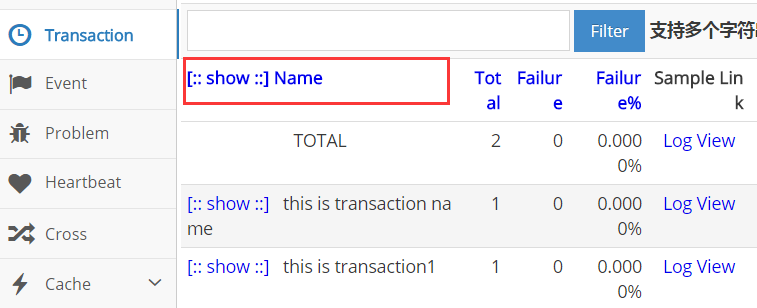
第二个参数为name。分别对应界面里的：

type和name不建议用中文

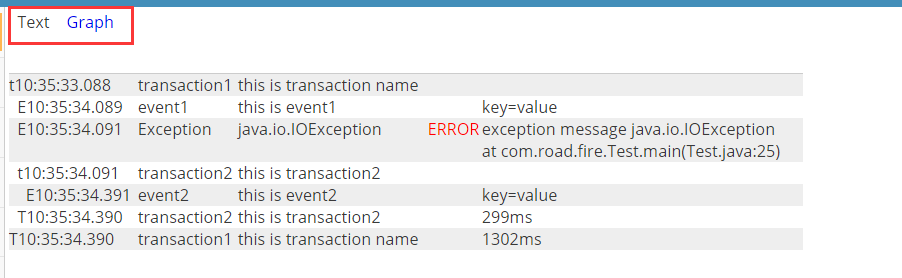
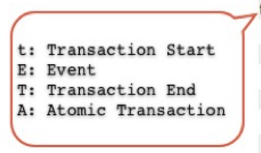
type



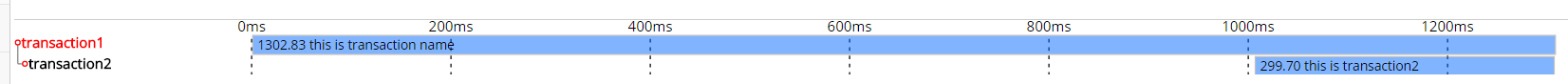
点击type可进入。首行对应name。



点击log View可查看整个Transaction的过程。如下：



点击Graph，可查看流程的执行时间



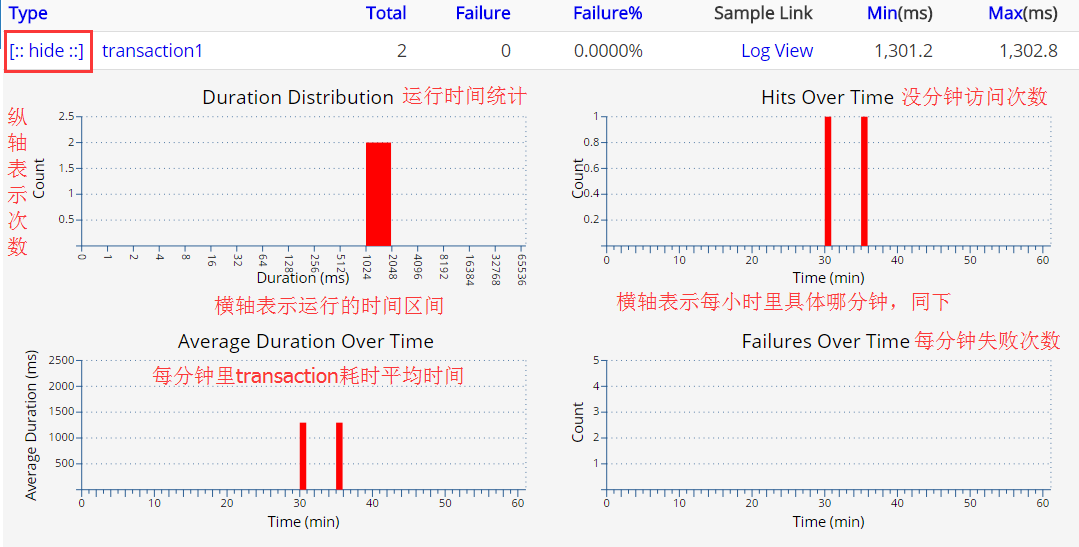
更多详情可查看：

http://unidal.org/cat/r/home?op=view&docName=user

注意：transaction只显示第一次埋点的信息。之后同样的埋点log View的内容不会变化。

transaction界面说明





总结：transaction埋点中间可以嵌套transaction，而其他埋点，如event， exception等的埋点都需要在transaction的过程中进行埋点才有效。每个transaction都需要调用transaction.complete();

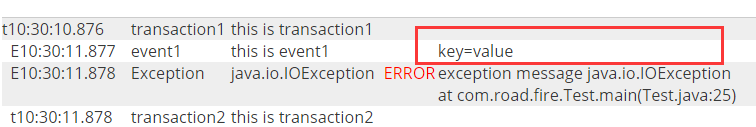
* 1. Event埋点

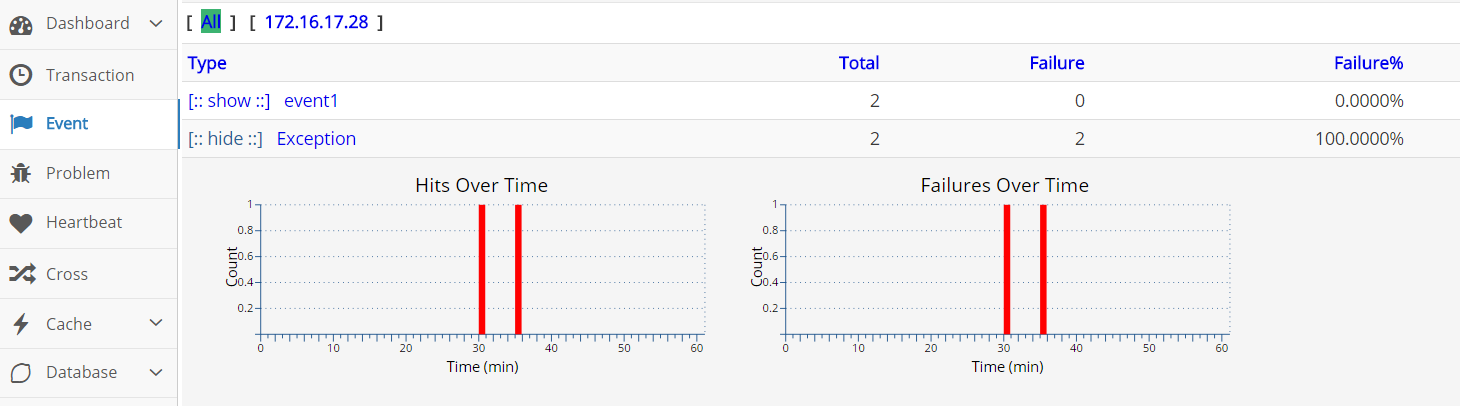
类似Transaction埋点，主要区别是Event埋点中没有执行时间统计。

用法：

Cat.*getProducer*().logEvent("type", "this is event name", Event.*SUCCESS*, "key=value");

最后一个参数对应logView里的某一参数。





详见文档：http://127.0.0.1:8080/cat/r/home?op=view&docName=user

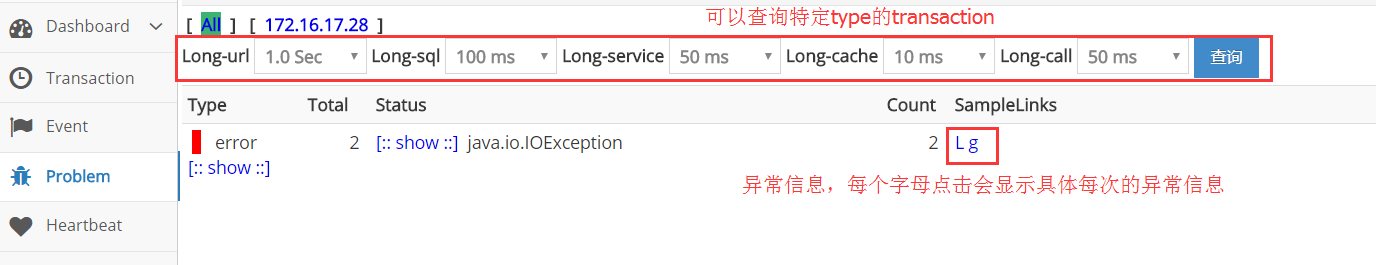
* 1. Exception埋点

使用方式：

Cat.getProducer().logError(**"exception message"**, **new** IOException());

第一个参数为说明消息，第二个为异常。

异常显示如下：



详情查看：http://unidal.org/cat/r/home?op=view&docName=user

* 1. 业务埋点

业务埋点有一下三种

Cat.logMetricForDuration("name", 1000);  
Cat.logMetricForSum("name", 3);  
Cat.logMetricForCount("name", 1);

1).logMetricForCount用于记录一个指标值出现的次数

2).logMetricForDuration用于记录一个指标出现的平均值

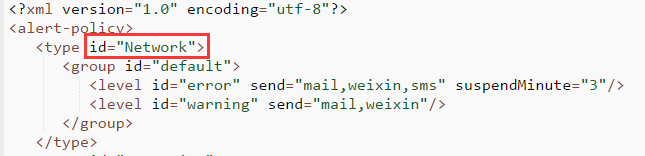
3).logMetricForSum用于记录一个指标出现的总和

统一粒度为一分钟。

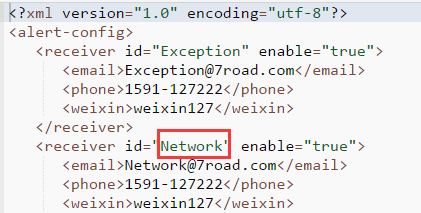
1. 告警设置
   1. 告警前提配置



告警策略默认已经配置好了。例子如下：



id对应默认告警人里的id。如下：



告警服务器默认已生成，只需做少许修改

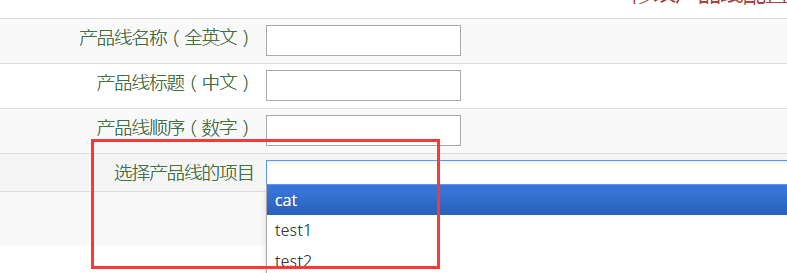


url需要自己配置成相对应的服务器请求地址，然后触发告警后会把<par>里的数据以键值对的参数传过去。

详情参考：http://unidal.org/cat/r/home?op=view&docName=alert

* 1. 业务告警设置

首先需要配置产品线。产品线项目里的是设置的appName

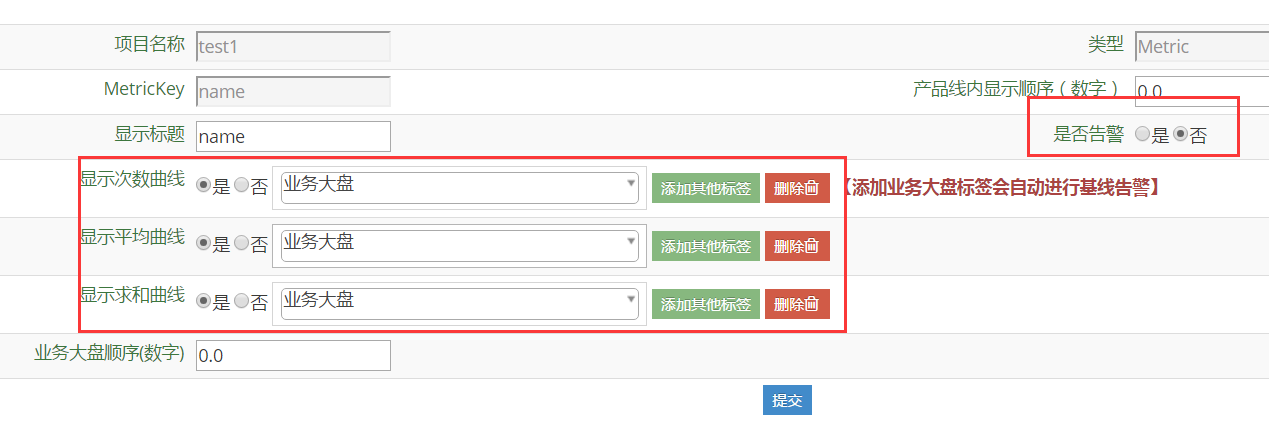
配置好后，就可以在业务监控里看到产品线。



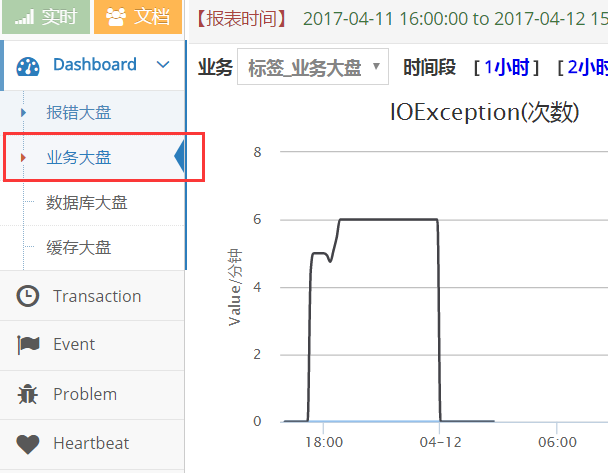
这时没有数据，只有调用了业务埋点后才会显示。如下:



之后可点击修改进行设置



若添加了曲线会显示在



然后点击告警按钮可以设置告警范围。当业务埋点的值超出此范围，则会发送消息给告警策略里 id=" Business "设置里的人。

详情查看链接里的业务告警：

http://unidal.org/cat/r/home?op=view&docName=alert 。

* 1. 异常告警

进行如下设置：



注意：异常名称要全名，域名为appName。异常名称可以Total，即所有异常。域名可为“Default”，即适用于所有项目组。

更多详情参考文档。其他告警设置类似，不一一说明。