**vmonitor-collector使用说明**

|  |  |
| --- | --- |
| **审核人** |  |
| **拟制人** | 刘政 |
| **提交日期** | 2014-02-25 |
| **更新日期** | 2014-04-29 |

|  |  |  |
| --- | --- | --- |
| **更新时间** | **更新人员** | **更新内容** |
| **20140411** | **刘政** | **增加Spring支持** |
| **20140429** | **刘政** | **增加Filter可配置支持** |

# 目录

[目录 3](#_Toc386541702)

[1 概述 3](#_Toc386541703)

[2 使用vmonitor-collector 4](#_Toc386541704)

[2.1 配置 4](#_Toc386541705)

[2.1.1 pom.xml 4](#_Toc386541706)

[2.1.2 logback.xml 5](#_Toc386541707)

[3 stargate客户端日志监控 5](#_Toc386541708)

[3.1 配置 5](#_Toc386541709)

[4 stargate服务端日志监控 6](#_Toc386541710)

[4.1 配置 6](#_Toc386541711)

[5 Http Filter日志监控 7](#_Toc386541712)

[5.1 配置 7](#_Toc386541713)

[5.2 使用 9](#_Toc386541714)

[6 Struts2 Interceptor日志监控 9](#_Toc386541715)

[6.1 配置 9](#_Toc386541716)

[6.2 使用 10](#_Toc386541717)

[7 Spring Interceptor日志监控 11](#_Toc386541718)

[7.1 配置 11](#_Toc386541719)

[7.2 使用 12](#_Toc386541720)

[8 custom-collector-config.properties文件 14](#_Toc386541721)

# 概述

本文档主要记录具体业务端如何使用vmonitor-collector提供的工具及服务。

# 使用vmonitor-collector

vmonitor-collector提供了stargate客户端及服务端的日志监控，它们都是和具体产品线无关的(简单理解为，在vmonitor-collector中不会出现产品线的名称)。

vmonitor最新相关文档及示例文件，可以下载vmonitor-collector-\*.\*.\*-RELEASE-resources.zip项目在src/doc下面找到，或者联系作者（[邮箱是liuzheng03@baidu.com](mailto:邮箱是liuzheng03@baidu.com))。

## 配置

### pom.xml

在你的pom.xml中加入如下依赖配置：

* 添加dependency依赖

<!-- star vmonitor log -->

<dependency>

<groupId>com.baidu.vmonitor</groupId>

<artifactId>vmonitor-collector</artifactId>

<version>1.2.0-RELEASE</version>

</dependency>

<!-- end vmonitor log -->

* 确保你的项目支持slf4j并保证可以打印出info级别信息，例如：

<!-- Log begin -->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

<version>${slf4j-version}</version>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>jcl-over-slf4j</artifactId>

<version>${slf4j-version}</version>

</dependency>

<dependency>

<groupId>ch.qos.logback</groupId>

<artifactId>logback-classic</artifactId>

<version>${logback-version}</version>

</dependency>

<!-- bridge log4j to logback -->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>log4j-over-slf4j</artifactId>

<version>${slf4j-version}</version>

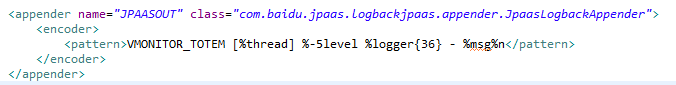
</dependency>

<!-- Log End -->

### logback.xml

在你的logback.xml中加入如下依赖配置：

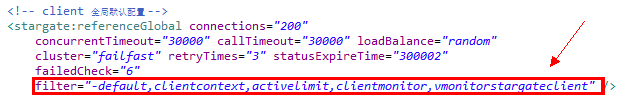
* 日志工具不一定是logback，只要是实现了slf4j规范的工具都可以，为了区分你的项目日志，请在日志打印内容格式前，加上“VMONITOR\_项目名”，例如：



# stargate客户端日志监控

## 配置

在你项目的stargate客户端配置中，加入如下配置即可：



<!-- client 全局默认配置 -->

<stargate:referenceGlobal connections=*"200"*

concurrentTimeout=*"30000"* callTimeout=*"30000"* loadBalance=*"random"*

cluster=*"failfast"* retryTimes=*"3"* statusExpireTime=*"300002"*

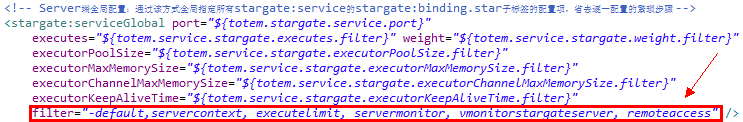
failedCheck=*"6"* filter=*"-default,clientcontext,activelimit,clientmonitor,vmonitorstargateclient"* />

如果你的stargate是1.1.14+版本，那么上面的filter你只需要：filter=*"vmonitorstargateclient"*即可，此处非常感谢stargate项目组大神刘涛提供的支持。

# stargate服务端日志监控

## 配置

在你项目的stargate服务端配置中，加入如下配置即可：



<!-- Server端全局配置：通过该方式全局指定所有stargate:service的stargate:binding.star子标签的配置项，省去逐一配置的繁琐步骤 -->

<stargate:serviceGlobal port=*"${totem.stargate.service.port}"* executes=*"${totem.service.stargate.executes.filter}"* weight=*"${totem.service.stargate.weight.filter}"* executorPoolSize=*"${totem.service.stargate.executorPoolSize.filter}"* executorMaxMemorySize=*"${totem.service.stargate.executorMaxMemorySize.filter}"* executorChannelMaxMemorySize=*"${totem.service.stargate.executorChannelMaxMemorySize.filter}"* executorKeepAliveTime=*"${totem.service.stargate.executorKeepAliveTime.filter}"* filter=*"-default,servercontext, executelimit, servermonitor, vmonitorstargateserver, remoteaccess"* />

如果你的stargate是1.1.14+版本，那么上面的filter你只需要：filter=*"vmonitorstargateserver"*即可，但是细心的你应该注意到了，Service比客户端多了一个remoteaccess，这个是处理服务端黑白名单的机制，目前stargate侧保留的默认顺序，这样会有一个问题，假如你想要记录被拒绝的访问时，配置filter=*"vmonitorstargateserver"*可能无法满足，此时你需要stargate插队机制解决，这样设置你的filter：filter=*"-remoteaccess,vmonitorstargateserver,remoteaccess"*，这样就可以记录被拒绝的访问，此处非常感谢stargate项目组大神刘涛提供的支持。

# Http Filter日志监控

## 配置

在你的项目web.xml中添加如下配置即可：

~~<!-- vmonitor log filter -->~~

~~<filter>~~

~~<filter-name>vmonitorLog</filter-name>~~

~~<filter-class>com.baidu.vmonitor.log.filter.LogFilter</filter-class>~~

~~</filter>~~

~~<filter-mapping>~~

~~<filter-name>vmonitorLog</filter-name>~~

~~<url-pattern>/\*</url-pattern>~~

~~</filter-mapping>~~

*（以上是1.1.0-release的配置，不支持自定义collectorClass与filter绑定）*

<!-- vmonitor log filter -->

<filter>

<filter-name>vmonitorLog</filter-name> <filter-class>com.baidu.vmonitor.log.filter.ConfigLogFilter</filter-class>

<init-param>

<param-name>collectorClass</param-name> <param-value>com.baidu.totem.web.action.TotemFilterCollector</param-value>

</init-param>

<init-param>

<param-name>key</param-name>

<param-value>protocol,consumingTime,method</param-value>

</init-param>

<init-param>

<param-name>behaviorType</param-name>

<param-value>DO\_LOG</param-value>

</init-param>

<init-param>

<param-name>value</param-name>

<param-value>true</param-value>

</init-param>

</filter>

<filter-mapping>

<filter-name>vmonitorLog</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

所有<init-param>标签都可以不配置，使用默认值，如需自定义可参照如下说明：

* collectorClass：日志收集者的class对象，默认使用vmonitor实现。如需自定义可实现FilterCollectorInterface接口并将实现类全类名配置在这个属性。
* key：日志信息的扩展key，默认是无扩展信息。collectorClass使用vmonitor实现 ，key可参考LogInfoKey，如自定义实现则按各自需求填写，以“,”分隔。
* behaviorType:是否使用vmonitor打印日志,默认是DO\_LOG，具体值可查看BehaviorType。
* value:是否启用这个Filter，默认启用。

配置的时候注意，假如你使用了类似Struts2等框架时，记得配置在相关框架filter之前，否则此filter无法起作用。

## 使用

默认记录的只有请求时间、请求URL、应答时间、是否成功。其他信息系统默认提供了一部分，需要你进行选择，选择方式是在你项目资源的根目录下，放置名为custom-collector-config.properties的文件，文件信息可以查看vmonitor最新相关文档及示例文件。

需要注意的是，由于使用了类似Struts2等框架，导致系统异常被拦截掉了，vmonitor的filter默认支持对请求是否成功的判断将失效，此时恐怕不能满足你的需要，你如果需要正确的判断，或者其它自定义方式的日志信息，那么你可以实现我们提供的FilterLogInfoCollectorInterface接口，在这里你将可以记录自定义请求和响应日志信息，具体实现方式可以查看vmonitor相关示例。

# Struts2 Interceptor日志监控

由于Http Filter的方式在某些框架的影响下，可能对请求是否成功的判断失效，对此特别提供了Struts2框架的日志监控功能。

## 配置

你只需要在你的Struts2配置文件中配置vmonitor的LogInterceptor即可，方式如下：

<interceptors>

<interceptor name=*"exception"* class=*"com.baidu.nmp.base.thirdparty.struts.ExceptionInterceptor"* />

<interceptor name=*"log"*

class=*"com.baidu.vmonitor.log.struts2.LogInterceptor"* />

<interceptor-stack name=*"myStack"*>

<interceptor-ref name=*"defaultStack"* />

<interceptor-ref name=*"exception"* />

<interceptor-ref name=*"log"* />

</interceptor-stack>

</interceptors>

<default-interceptor-ref name=*"myStack"* />

值得特别注意的是，LogInterceptor要配置在最后一个，否则按照Struts2的interceptor执行原理，可能直接避开了应答日志监控程序。

## 使用

默认记录的只有请求时间、请求URL、应答时间、是否成功。其他信息系统默认提供了一部分，需要你进行选择，选择方式是在你项目资源的根目录下，放置名为custom-collector-config.properties的文件，文件信息可以查看vmonitor最新相关文档及示例文件。

你如果需要其它自定义方式的日志信息，那么你可以实现我们提供的Struts2LogInfoCollectorInterface接口，在这里你将可以记录自定义请求和响应日志信息，具体实现方式可以查看vmonitor相关示例。

# Spring Interceptor日志监控

## 配置

1. 使用注解的方式，精确的决定某一个方法是否打印日志（意思就是该方法有@DoLog注解才打印日志），则需要使用的是DoLogDefaultInterceptor，在你的Spring.xml中加入如下依赖配置：

<aop:aspectj-autoproxy proxy-target-class=*"true"* />

<bean class=*"com.baidu.vmonitor.log.spring.DoLogDefaultInterceptor"* />

1. 使用注解的方式，决定某一个方法是否打印日志（意思就是该方法、方法的类、父类、接口其中之一有@DoLog注解就打印日志），则需要使用的是DoLogExpandInterceptor，在你的Spring.xml中加入如下依赖配置：

<bean id=*"doLogExpandInterceptor"* class=*"com.baidu.vmonitor.log.spring.DoLogExpandInterceptor"* />

<aop:config>

<aop:aspect id=*"doLog"* ref=*"doLogExpandInterceptor"*>

<aop:pointcut id=*"allMethod"*

expression=*"execution(\* com.baidu.totem.manager..\*.\*(..))"* />

<aop:around method=*"aroundAdvice"* pointcut-ref=*"allMethod"* />

</aop:aspect>

</aop:config>

1. 如果不想使用注解和自定义日志，只想简单配置切面，则需要使用的是DefaultInterceptor在你的Spring.xml中加入如下依赖配置：

<bean id=*"defaultInterceptor"* class=*"com.baidu.vmonitor.log.spring.DefaultInterceptor"* />

<aop:config>

<aop:aspect id=*"doLog"* ref=*"defaultInterceptor"*>

<aop:pointcut id=*"allMethod"*

expression=*"execution(\* com.baidu.totem.manager..\*.\*(..)) or execution(\* com.baidu.totem.service..\*.\*(..)) or execution(\* com.baidu.totem.dao..\*dao.\*(..))"* />

<aop:around method=*"aroundAdvice"* pointcut-ref=*"allMethod"* />

</aop:aspect>

</aop:config>

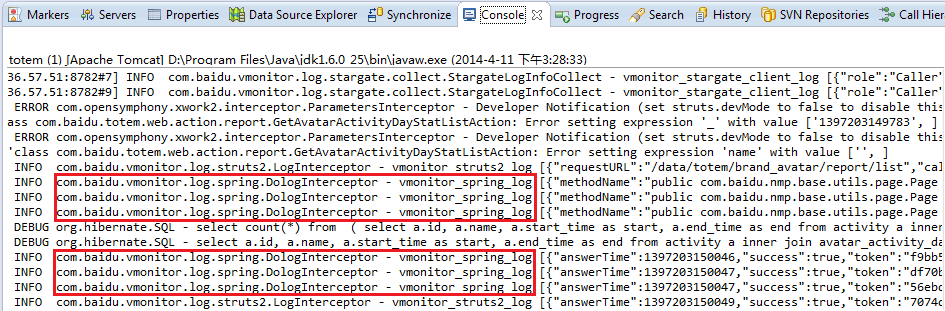
1. 如果不想使用注解却需要自定义打印日志，则自己根据Spring AOP实现，不需要vmoniter的Spring支持。

需要注意的是：

* 第一二种注解方式支持自定义日志打印，第三种则使用默认的方式打印日志。
* 在使用注解或者xml配置切面的时候，注意只有托管给Spring的类才可以使用，也就是说类似Struts2的action、传统的HttpServlet不能使用这种方式，否则会引起异常（可参考Http Filter及Struts2 Interceptor解决方案）。
* 作者推荐使用第二种，自定义切面，注解支持继承，可以将注解都写在接口上，减少实现类的注解量。例如Action、Service、Dao各自实现一个标记型注解（可参照Serializable、Cloneable），将@DoLog标记在自定义的标记型注解上，达到统一管理配置的目的。后期有必要的话，作者会安排时间实现XML配置，此项待定，低优先级。

## 使用

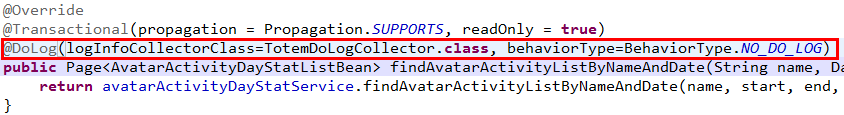
在你需要记录日志的方法上加上@DoLog注解即可；



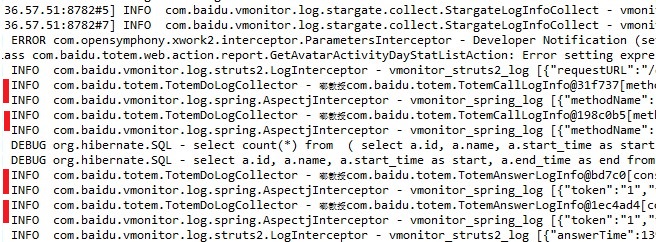
默认记录的只有请求方法、请求时间、应答时间、是否成功。其他信息系统默认提供了一部分，参照LogInfoKey.java需要你进行选择，选择方式参照@DoLog注解，注解影响范围配置（custom-collector-config.properties）可以查看vmonitor最新相关文档及示例文件。

你如果需要其它自定义方式的日志信息，那么你可以实现我们提供的DoLogCollectorInterface接口，在这里你将可以记录自定义请求和响应日志信息，具体实现方式可以查看vmonitor相关示例。

使用自定义日志打印方式需要配置注解参数，例如：



logInfoCollectorClass是你自定义日志打印的类，behaviorType是告诉vmonitor的日志打印行为，默认是帮助你打印信息，由于我的TotemDoLogCollector自己打印了日志，所以不想vmonitor重复打印一遍，故配置成NO\_DO\_LOG，否则打印两次如：



# custom-collector-config.properties文件

custom-collector-config.properties文件可参考如下附件：



#log

~~#filter请求日志扩展属性可选择配置serviceName,serviceIP,servicePort,clientIP,clientPort,protocol,method,Parameters~~

~~vmonitor.collector.log.filter.call.info=serviceName,serviceIP,servicePort,Parameters~~

~~#filter应答日志扩展属性可选择配置exceptionMessage,consumingTime~~

~~vmonitor.collector.log.filter.answer.info=~~

~~#filter请求应答自定义日志记录收集者类 ，类全路径。例如：com.baidu.totem.vmonitor.TotemCollector~~

~~vmonitor.collector.log.filter.collector=com.baidu.totem.vmonitor.TotemCollector~~

*（以上是1.1.0-release的配置，不支持自定义collectorClass与filter绑定）*

#struts2请求日志扩展属性可选择配置serviceName,serviceIP,servicePort,clientIP,clientPort,protocol,method,className,methodName,Parameters

vmonitor.collector.log.struts2.call.info=serviceName,serviceIP,servicePort,className,methodName,Parameters

#struts2应答日志扩展属性可选择配置exceptionMessage,consumingTime,result

vmonitor.collector.log.struts2.answer.info=exceptionMessage

#struts2请求应答自定义日志记录收集者类 ，类全路径。例如：com.baidu.totem.vmonitor.Struts2Collector

vmonitor.collector.log.struts2.collector=com.baidu.totem.vmonitor.Struts2Collector

#请求应答日志记录级别，仅支持TRACE、DEBUG、INFO、WARN、ERROR五种，缺省为INFO

vmonitor.collector.log.level=INFO

vmonitor.collector.log.stargate.level=

vmonitor.collector.log.filter.level=

vmonitor.collector.log.struts2.level=

vmonitor.collector.log.spring.level=

如果看不到可在vmonitor-collector-\*.\*.\*-RELEASE-resources.zip包src/doc中找到或者联系liuzheng03@baidu.com