监控平台软件使用说明书

版本：0.0.1-SNAPSHOT

# 服务端

## 启动方式

服务端为一个springboot写的web项目，使用命令：java -jar monitoring-server-web-0.0.1-SNAPSHOT.jar，直接启动。

## 配置文件说明

项目包含一个“monitoring.properties”的配置文件，用于配置监控属性。具体说明如下：

|  |  |  |
| --- | --- | --- |
| 配置项 | 描述 | 默认值 |
| **monitoring.threshold** | *判断注册上来的应用实例是否离线的监控阈值，分为1到10共十级，表示超过多少个心跳包没收到就算离线* | **5** |
| **monitoring.alarm.enable** | *监控告警是否打开* | **true** |
| **monitoring.alarm.type** | *监控告警方式，有短信(***SMS***)和邮件(***EMAIL***)两种方式，邮件方式暂未实现* |  |
| **monitoring.alarm.sms.phone-numbers** | *手机号码，多个号码用英文逗号隔开* |  |
| **monitoring.alarm.sms.address** | *短信接口地址* |  |
| **monitoring.alarm.sms.protocol** | *短信接口协议* |  |
| **monitoring.alarm.sms.enterprise** | *短信接口提供商，因为不同的接口提供商可能有不同的接口调用方式* |  |

# 代理端

## 启动方式

代理端为一个springboot写的web项目，使用命令：java -jar monitoring-agent-0.0.1-SNAPSHOT.jar，直接启动。

## 配置文件说明

项目包含一个“monitoring.properties”的配置文件，用于配置监控属性。具体说明如下：

|  |  |  |
| --- | --- | --- |
| 配置项 | 描述 | 默认值 |
| **monitoring.server.url** | *监控服务端url* |  |
| **monitoring.server.username** | *监控服务端用户名* |  |
| **monitoring.server.password** | *监控服务端密码* |  |
| **monitoring.own.instance.name** | *实例名称，为英文名称* |  |
| **monitoring.heartbeat.rate** | *与服务端发心跳包的频率（秒）* |  |
| **monitoring.server-info.rate** | *与服务端发服务器信息包的频率（秒）* |  |

# 客户端

客户端为一个用Java写的插件，只适用于Java项目，并且针对SpringBoot和SpringMVC提供了专门的快速集成和使用方式。

## 配置文件说明

要使用监控客户端，必须在项目中添加一个监控属性文件，具体配置项说明如下：

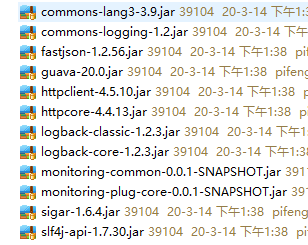
|  |  |  |
| --- | --- | --- |
| 配置项 | 描述 | 默认值 |
| **monitoring.server.url** | *监控服务端或代理端url* |  |
| *monitoring.server.username* | *监控服务端或代理端用户名* |  |
| *monitoring.server.password* | *监控服务端或代理端密码* |  |
| **monitoring.own.instance.name** | *实例名称，为英文名称* |  |
| **monitoring.heartbeat.rate** | *与服务端或者代理端发心跳包的频率（秒）* | **30** |
| **monitoring.server-info.rate** | *与服务端或者代理端发服务器信息包的频率（秒）* | **60** |

## 集成方式

### 未使用项目管理工具的Java项目

#### 添加jar包

对于未使用项目管理工具的Java项目，要使用监控客户端，需要添加如下jar包：



其中，monitoring-common-0.0.1-SNAPSHOT.jar和monitoring-plug-core-0.0.1-SNAPSHOT.jar两个jar包为监控jar包，其余的为监控所依赖的工具包，工具包版本可根据项目自行选择。

#### 启用监控功能

在main方法中，调用监控功能开启方法**Monitor.start()**来开启监控功能，或者调用重载的方法**Monitor.start(configPath, configName)**来指定监控配置文件的路径和名字从而开启监控功能，如果未指定配置文件路径和名字，配置文件需要放在classpath根路径下，名字必须为“monitoring.properties”。

### 使用项目管理工具的Java项目

#### 添加依赖

a.如果是maven项目，需要在需要在pom.xml文件中添加如下依赖：

<dependency>

<groupId>com.transfar</groupId>

<artifactId>monitoring-plug-core</artifactId>

<version>0.0.1-SNAPSHOT</version>

</dependency>

b.如果是gradle项目，需要在build.gradle文件中添加如下依赖：

compile group: 'com.transfar', name: 'monitoring-plug-core', version: '0.0.1-SNAPSHOT'

#### 启用监控功能

在main方法中，调用监控功能开启方法**Monitor.start()**来开启监控功能，或者调用重载的方法**Monitor.start(configPath, configName)**来指定监控配置文件的路径和名字从而开启监控功能，如果未指定配置文件路径和名字，配置文件需要放在classpath根路径下，名字必须为“monitoring.properties”。

### SpringBoot项目

#### 添加依赖

a.如果是maven项目，需要在需要在pom.xml文件中添加如下依赖：

<dependency>

<groupId>com.transfar</groupId>

<artifactId>monitoring-plug-core-spring-boot-starter</artifactId>

<version>0.0.1-SNAPSHOT</version>

</dependency>

b.如果是gradle项目，需要在build.gradle文件中添加如下依赖：

compile group: 'com.transfar', name: 'monitoring-plug-core-spring-boot-starter ', version: '0.0.1-SNAPSHOT'

#### 启用监控功能

在项目启动类上加上注解：@EnableMonitoringPlug来开启监控功能，或者通过注解的两个参数来指定配置文件的路径和名字，如果未指定配置文件路径和名字，配置文件需要放在classpath根路径下，名字必须为“monitoring.properties”。

### SpringMVC项目

#### 添加jar包

如果springmvc项目没有使用项目管理工具，那么在2.1.1节所列举的jar基础上，还需要添加一个jar包：monitoring-plug-core-spring-mvc-integrator-0.0.1-SNAPSHOT.jar。

如果使用了项目管理工具：

1. maven项目：

<dependency>

<groupId>com.transfar</groupId>

<artifactId> monitoring-plug-core-spring-mvc-integrator-0.0.1-SNAPSHOT</artifactId>

<version>0.0.1-SNAPSHOT</version>

</dependency>

1. gradle项目：

compile group: 'com.transfar', name: ' monitoring-plug-core-spring-mvc-integrator-0.0.1-SNAPSHOT ', version: '0.0.1-SNAPSHOT'

#### 启用监控功能

在web.xml文件中配置一个监听器，来开启监控功能：

<context-param>

<param-name>configLocation</param-name>

<param-value>classpath:monitoring.properties</param-value>

</context-param>

<listener>

<listener-class>

com.transfar.plug.listener.MonitoringPlugInitializeListener

</listener-class>

</listener>

## 发送自定义告警

项目只要集成了监控客户端，就能使用主动发送自定义监控信息的功能。具体使用方式如下：

在需要发送告警信息的位置，创建一个Alarm对象 ，添加告警信息，再调用发送告警信息的方法，就可发送自定义告警到监控服务端。例如：

**Alarm alarm = Alarm.*builder*().alarmLevel(AlarmLevelEnums.*INFO*).title("测试").charset(Charsets.*UTF\_8*).msg("测试监控客户端！").build();**

**Monitor.sendAlarm(alarm);**