# 一、程序功能描述

本程序读取指定格式日志的日志文件，并将其中含有异常信息的日志条目提取出来保存在单独的文件，最后将统计不同种类异常的数量，并将统计的结果打印到excel文件中，并存放到指定的目录中。

## 1.提取的异常日志文件：

程序会在源日志文件的目录中创建名为<源日志文件名-exception.log> 的新日志文件，新创建的异常日志文件中将只含有源日志文件中包含异常信息的日志条目，同时在每条异常日志的前面添加了其原始的日志文件来源，以及在原始日志文件中的行号，方便查找原始异常日志的上下文信息，以进行后续的异常问题定位。

比如如下一个日志文件，是原始的全量日志

server.log

而运行后生成这个文件，是提取后的异常日志

server.log-exception.log

## 2.统计的excel表格

程序会在指定的目录中生成一个excel文件，其文件名为 exception statistic-运行时间戳.xlsx，该文件含有按照一定匹配长度计算出来的相同异常日志的数量信息，其结果是对于数量 那一列进行降序排序后的结果，方便查找反复出现的异常日志。

如下所示：

exception statistic-20190908181212.xlsx

# 二、生成文件格式说明

## 1.异常日志文件的格式：

Original log file absolute path: C:\Users\JasonFitch\Desktop\logfetch\0906\guanli\server.log

Original log entry line number : 104

####|2019-08-19 22:22:15.869|SEVERE|org.apache.catalina.core.StandardContext|\_ThreadID=19;\_ThreadName=pool-1-thread-10;\_RequestID=a73e857e-aec7-4a35-aaa7-b7e4857154a0;|WEBTIER101270: Exception starting filter AuthFilter

java.lang.NoClassDefFoundError: org/apache/log4j/Logger

at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)

……ignore some tracestack……

……ignore some tracestack……

……ignore some tracestack……

at java.lang.Thread.run(Thread.java:662)

Caused by: java.lang.ClassNotFoundException: org.apache.log4j.Logger

at org.apache.catalina.loader.WebappClassLoader.loadClass(WebappClassLoader.java:1620)

... 22 more

|####

格式说明：

第一行：该异常日志所属的原始日志文件，用来查找原始的文件。

第二行：该异常日志在原始日志文件中行号，用来查找原始异常位置，并定位其异常上下文信息。

第三行：该异常日志的原始内容，用来查看异常的具体细节。

第四行：空行，用来分隔不同的异常日志条目。

## 2.Excel文件统计格式：

Exception Key Occur Count

ResTypeMgrService refresh end.CacheStats{hitCount= 134

DictItemService refresh failed:org.springframework 89

DeskSoftServiceImpl.querySysParam Exception: while 84

com.huawei.boss.inventory.common.ReceptionExceptio 13

WEBTIER101418: Method service of Servlet action in 3

格式说明：

Exception Key： 按照参数 -n 指定的匹配长度截取的数据

Occur Count ： 相同Exception Key在原始日志文件中出现的次数。

Excel的输出结果已经按照每种异常的数量进行了降序排序，方便查找大量出现的异常，

同时这个excel文件中可能会含有多个sheet，当在使用 -f 参数传递多个文件到程序时，所有的统计信息都会打印在同一个excel中，但是每一个文件统计的信息会独立的打印在一个单独的sheet中。可以通过切换不同的sheet标签来查看不同文件的日志统计问信息。

## 3.使用建议：

可以通过看excel左下角的sheet标签来得到该异常所在的文件，然后在相应的异常日志文件中搜索，相应的 Exception Key 部分，然后更具异常日志文件中记录的原始行号找见异常发生的上下文信息从而定位问题。

# 二、程序帮助信息

通过直接运行命令可以得到如下帮助信息

C:\Users\JasonFitch\Desktop\logfetch\target>java -jar logfetch.jar

Missing required option: f

usage: bootstrap.LogAnalyzerMain

-d,--out-dir <arg> optional, result output dir, default <C:\Users\JasonFitch\Desktop\logfetch\target> .

-e,--log-encoding <arg> optional, log file encoding, default <GB2312> .

-f,--log-files <arg> input files, comma separated list of input files.

-n,--match-length <arg> optional, character length that matching the same exception, default <50> .

-t,--log-type <arg> optional, log file type, currently supports bes and weblogic weblogic2 logs, default <bes> .

## 输入参数说明:

-t 可选的，解析的日志格式, 默认值： bes

-e 可选的，解析的日志编码， 默认值： GB2312

-n 可选的，相同日志匹配长度， 默认值： 50

-d 可选的，Excel统计文件的输出位置 默认值： 当前程序工作目录（依照运行环境的变化而变化）

-f 必要的，要解析的日志文件列表。 默认值： 无

# 三、实际运行效果：

执行如下命令可以看见程序的实际运行效果：

C:\Users\JasonFitch\Desktop\logfetch\target>java -jar logfetch.jar -f C:\Users\JasonFitch\Desktop\logfetch\0906\guanli\server.log

user.dir : C:\Users\JasonFitch\Desktop\logfetch\target

Analyzer runtime arguments info:

log-type : bes

log-encoding: GB2312

match-length: 50

Target files that may be processed:

C:\Users\JasonFitch\Desktop\logfetch\0906\guanli\server.log

Excel statistic result file path

C:\Users\JasonFitch\Desktop\logfetch\target

Target log file C:\Users\JasonFitch\Desktop\logfetch\0906\guanli\server.log exist, process it!

create exception info file C:\Users\JasonFitch\Desktop\logfetch\0906\guanli\server.log-exception.log success!

create excel file C:\Users\JasonFitch\Desktop\logfetch\target\exception statistic-20190908181212.xlsx

## 输出信息说明:

1.程序工作目录

2.程序执行参数

3.程序将要处理的文件

4.程序统计结果实际目录

5.处理的文件信息，创建的异常日志文件信息，统计结果文件信息

# 四、关于日志格式的说明

可以通过 -t 参数来指定要解析的日志格式，目前不支持混合格式解析，目前可以解析BES和weblogic的如下日志格式，如果有其他的格式需要解析，可以通过增加相应类来填加可以解析的格式，如果提供错误的格式参数，那么程序将会拒绝解析。

## 1.参数为 bes 时解析的格式

bes日志格式 标准模版

####| time | level | module | thread | detail |####

## 2.参数为 weblogic 时解析的格式

weblogic日志格式 第一种模版

< time > < level > < module > < code > < info > detail

## 3.参数为 weblogic2 时解析的格式

weblogic日志格式 第二种模版

< time > < level > < module > < app > < file > < thread > < source > < blank1 > < blank2 > < timestamp > < code > < detail >

# 五、扩展自定义格式的方法

需要实现的类有：

entry.AbstractLogRecord

用于结构化存储日志

parser.AbstractLogParser

用于解析日志，按照自定义格式将日志由文本转变为结构化JAVA对象

interceptor.AbstractInterceptor

用于处理日志，比如数量统计，异常提取之类的

analyzer.AbstractAnalyzer

用于整合日志的解析逻辑和处理逻辑

最后在启动类bootstrap.LogAnalyzerMain的bootstrap.LogAnalyzerMain#invokeAnalyzer 方法中添加新增加的解析格式便可。

# 六、编译结果

本工程是maven工程,可以直接使用mvn clean install 将程序编译并放到本地库，结果会生成俩个jar包，这俩个jar包都是可执行的。

## 1.logfetch.jar

第一个不含依赖，需要配合编译目录下的lib目录中的依赖使用，manifest中已经添加了lib下的依赖包作为类路径

## 2.logfetch-jar-with-dependencies.jar

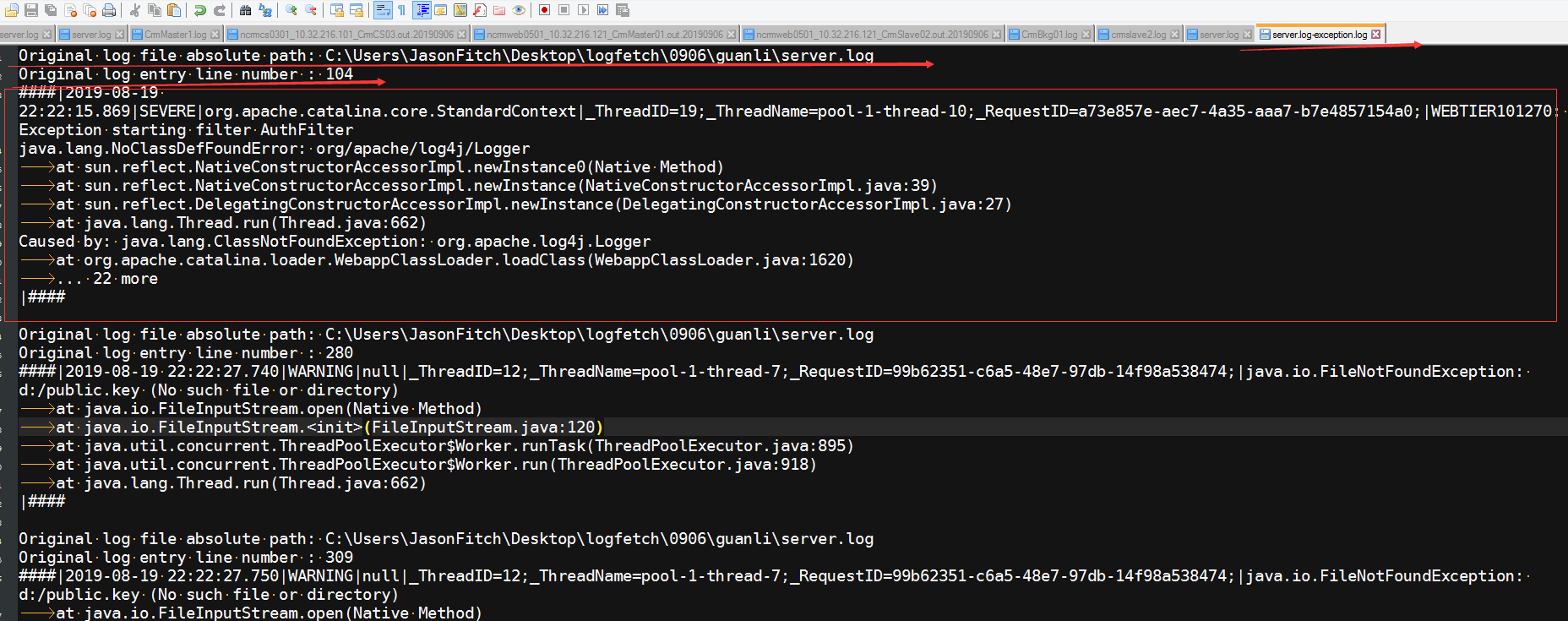
这个jar包是包含依赖的，fat-jar,可以做到单文件运行，依赖已经解包并和应用的类放到了一起了。

# 七、缺陷

对于日志开头含有时间部分的日志，由于时间不同会被解析成为不同的异常，这个应该可以通过替换掉这些时间部分后在匹配，或者是使用更准确的匹配方式来处理，暂时还没有这样处理。

# 八、附录

## 1.日常日志文件图



## 2.统计Excel文件图

