

LOG FOUINEUR

Code specifications

A log parser (System logs, application logs, servers logs ...)

Reference : JLP/LogFouineur/2015

| Version | Date | Comments |
|---------|---------|-----------------|
| V1.0 | 12-2015 | Initial Version |
| | | |
| | | |

Table des matières

| | | |
|---------|---|---|
| 1 | Introduction..... | 3 |
| 1.1 | History..... | 3 |
| 1.2 | Why “LogFouineur” ?..... | 3 |
| 1.3 | General conception..... | 3 |
| 2 | Stream records creation..... | 4 |
| 2.1 | Definition of Value Class and Pivot Class..... | 4 |
| 2.2 | Extracting the date..... | 4 |
| 2.2.1 | Case of explicit date..... | 4 |
| 2.2.2 | Case of implicit date..... | 5 |
| 2.2.2.1 | Implicit date and timestamped records..... | 5 |
| 2.2.2.2 | Implicit date and records separated with a time step..... | 5 |
| 2.3 | Extracting Values..... | 5 |
| 2.4 | Extracting Pivots..... | 6 |
| 2.5 | All together..... | 7 |
| 3 | User Interface..... | 7 |
| 3.1 | Create a new project..... | 7 |
| 3.2 | Open an existing project..... | 8 |
| 3.3 | Adding or modifying a scenario in an opened project..... | 9 |

1 Introduction

1.1 History

At work, I have developed, in java + scala, a log parser for parsing text files where logs are timestamped (with human readable date, or in milliseconds associated with an explicit or implicit start date).

This development was done in “fast and dirty” way. I know that there are a lot of misconception. The last version, I made is accessible at github,

<https://github.com/PASTJL/swingScaViewer>

Unfortunately, this tool can't evolve (only compilation with Java 1.6 is suitable) because there are incompatibilities between Scala version and new Java Version, specially with Swing API. So, I decide to entirely re-write this tool with :

- Java 8 / Streams / Lambdas expressions
- JFX for replacing Swing interface
- Disruptor API for handling concurrency

<http://mvnrepository.com/artifact/com.lmax/disruptor/3.3.2> . This version can evolve, a 3.3.3 is yet available in github repository: <https://github.com/LMAX-Exchange/disruptor>

- JFreeChart for charting <http://mvnrepository.com/artifact/org.jfree/jfreechart/1.0.19>

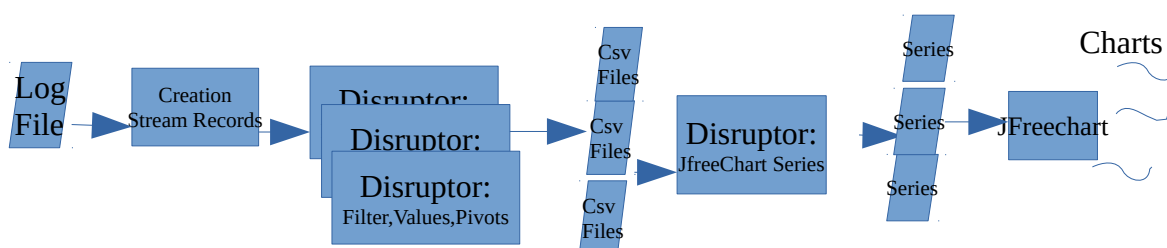
1.2 Why “LogFouineur” ?

Because all English words, for this logs analysis (scan, parser, browser ...) are already used.

“ Une fouine ” in French is a weasel :



1.3 General conception



2 Stream records creation

The input is a text file that can be gzipped.

The file is read with a stream, and we apply Lambdas, to create records (that can be constituted by multiple lines). This first chain is mono-threaded. A basic filter is done to include / exclude records. The filters are Pattern Matching based with the java Pattern class.

The output is a Stream of records that will be the Producer of the first Disruptor Pattern.

The generation of these records is optimized for performance, so there are several identical methods in the class RecordReader, to avoid a lot of tests when reading the log file.

A record is detected by 2 regex patterns, beginRegex and endRegex, there are three cases :

- when the beginRegex is only defined (endRegex = null), the records are the line of the input log file.
- when the beginRegex and the endRegex are both defined and the 2 regex are different, the records are constituted with several consecutive lines, including the production of the endRegex.
- when the beginRegex and the endRegex are both defined and the 2 regex are equal, the records are constituted with several consecutive lines, excluding the production of the endRegex. The production of the endRegex is conserved to begin the next record. Some definitions and challenges

2.1 Definition of Value Class and Pivot Class

In a record, created as seen above, we have to extract :

- a date that must be the first column of the csv file
- extract at least a Value, but also several Values can be also been extracted
- extract Pivots (none Pivot is also a possibility) that allow to group values => example a duration (value) may be analysed grouped by URL (pivot).

2.2 Extracting the date

2.2.1 Case of explicit date

That means that the date, in the record, is human readable (like in Apache access logs, Tomcat access Logs, Log4J logs ...).

```
10.186.9.87 - - [12/May/2010:15:41:14 +0200] 1949759 "POST /krmx69/es HTTP/1.1" 200 526
10.170.213.100 - - [12/May/2010:15:41:14 +0200] 2247214 "POST /krmx69/es HTTP/1.1" 200 526
```

To extract the date from the record, a Java Pattern Regex is used. If a regex is not sufficient, we can use 2 regex separated by a global separator that is not part of the regex. This separator must be carefully chosen . It is defined in the **config/logFouineur.properties** and it can be changed if

needed.

When the date is extracted, it must be translated to a Java Date . For the Date class **SimpleDateFormat** does the job.

The records of this kind of file may be treated in concurrent manner if values allow it.

2.2.2 Case of implicit date

Two sub-cases :

- the records are timestamped (in milliseconds, or seconds or other ...) relatively to a start date
- no time-stamp in the record, but the step between 2 records is known, and the start date is also known.

2.2.2.1 *Implicit date and timestamped records*

The records are timestamped (in milliseconds, or seconds or other ...) relatively to a start date. This start date must be the epoch date.

We use the same principle to extract the number with a regex, no Java Date Format is needed. We must only the unit of the time-stamp and the start date, and we can create a Java Date Class.

The records of this kind of file may be treated in concurrent manner if values allow it .

2.2.2.2 *Implicit date and records separated with a time step*

The records are not timestamped, but between two records we know the time gap. The start date must be known.

Example of such files : output of a **vmstat** in a Unix box.

The records of this kind of file must be treated with a unique Java Thread.

2.3 Extracting Values

| Value |
|--|
| - String Name - String strPattern1 - String strPattern2 - String unit - double scale |

Above, the Class Value with the principle attributes.

A record must have at least one Value.

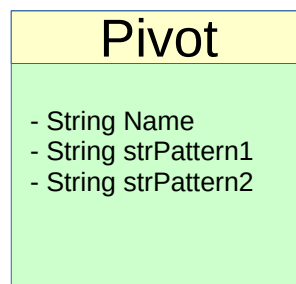
There are two ways to extract Values:

- the most common and recommended : by using Java Pattern Matching on the record, in two phases if necessary. If it is impossible to extract the value with the first regex, first production of strPattern1 is matched against strPattern2 to extract the final value.
- If the way above is not possible : by using customized plugin with a specific Java Class. The strPattern1 contains the chain **plugin=<ClassName of the Plugin>**, and strPattern2 contains parameters separated by the global separator. Later in the document, I will explain how to do.

The unit defines the unit of the value inserted in the csv file, and it is possible to change the scale of the unit to translate the value (ex μ s \rightarrow ms).

Value are handled with the primitive double Java, when inserted in csv file.

2.4 Extracting Pivots

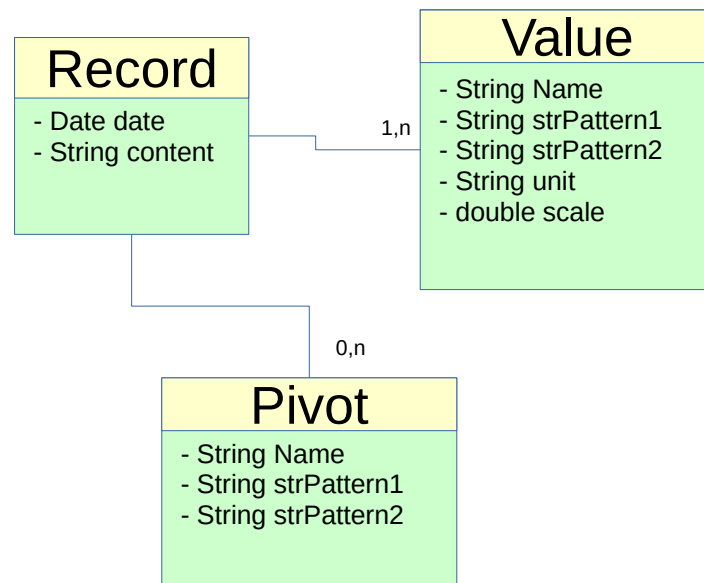


Above, the Class Pivot with the principle attributes.

A record may have 0 to n Pivots

The way to extract Pivots : by using Java Pattern Matching on the record, in two phases if necessary. If it is impossible to extract the pivot with the first regex, first production of strPattern1 is matched against strPattern2 to extract the final pivot.

2.5 All together



3 User Interface

The choice of UI is the JFX Framework. I decide to use the pure Java programming way (not FXML / SceneBuilder) to better understand how JFX runs.

3.1 Create a new project

To create a new analysis project. Most often, It is related to a version of an application.

The screenshot shows the 'Create a new project' dialog box in LogFouineur. The dialog has a title bar 'LogFouineur : Create a New Project' and a main title 'Create a new project'. It contains several input fields and buttons:

- Existing projects**: A dropdown menu showing 'jpjp'.
- New project**: A text input field with placeholder text 'Fill with the name of project'.
- Scénario name**: A text input field with placeholder text 'Fill with the name of scenario'.
- Begin Date**: A text input field with placeholder text 'Begin date of the scenario' and format 'yyyy/MM/dd:HH:mm:ss'.
- End Date**: A text input field with placeholder text 'End date of the scenario' and format 'yyyy/MM/dd:HH:mm:ss'.
- Create** and **Cancel** buttons at the bottom.

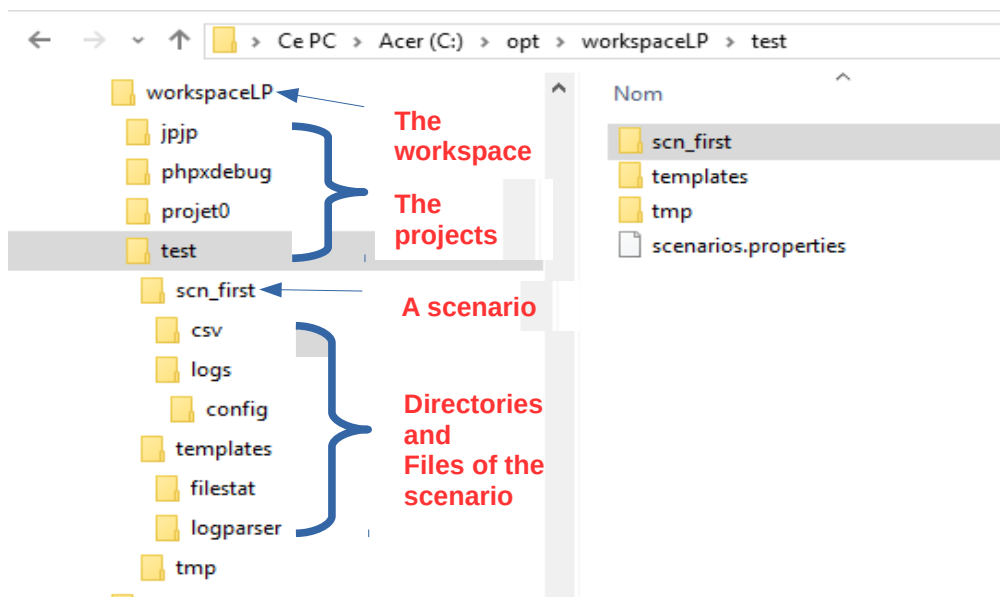
New project :

- must have a name different than existing projects

- name length must be greater than 3.
- can specify the name of the first scenario (if no name, “default” is chosen)
- can specify the beginning and the end of the first scenario in respect with the indicated format. If default, Begin Date=1970/01/01:00:00:00, End Date= <the current date => new Date();>

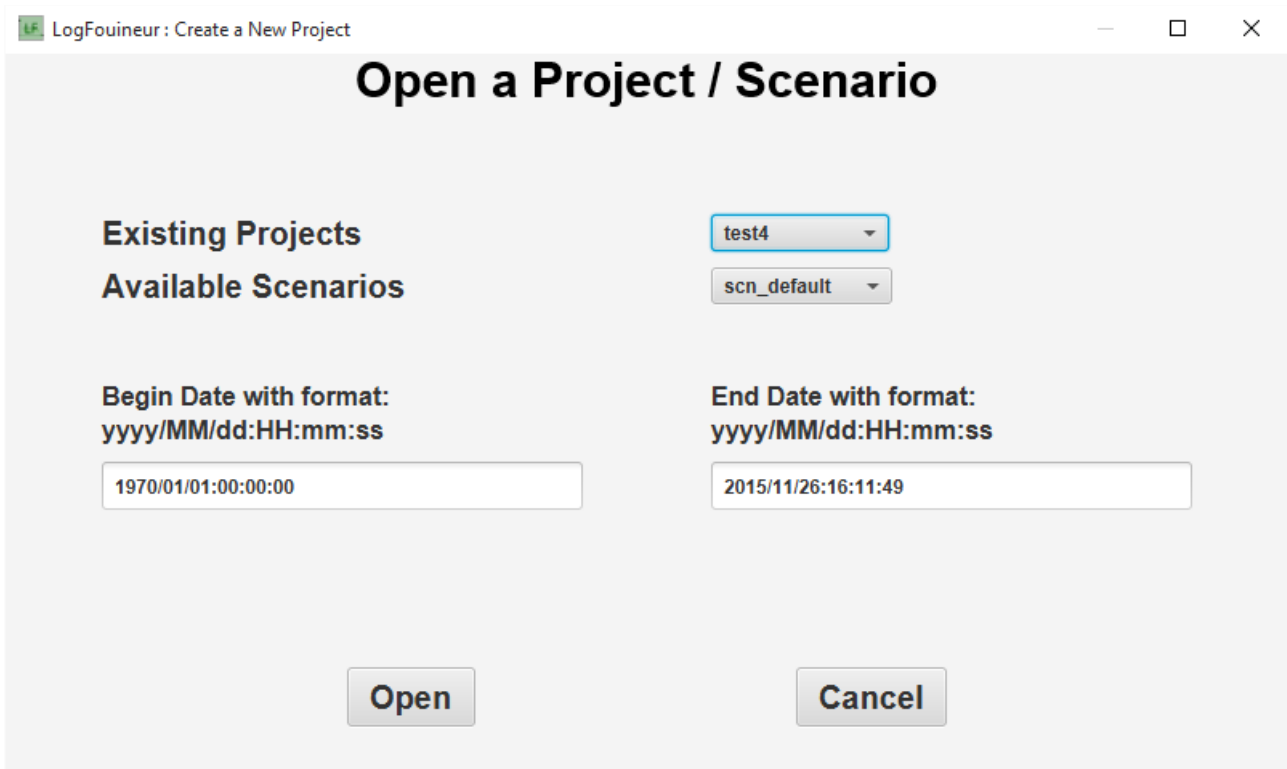
These dates allow to parse the logs file into two interesting dates and increase the performance of parsing when the logs files cover a very long interval.

The organization of the repositories are shown below :



3.2 Open an existing project

After clicking File → Open Project, you get a Dialog as shown below :



The screenshot shows a dialog box titled "Open a Project / Scenario" with a standard Windows-style title bar. Inside the dialog, there are two sections: "Existing Projects" and "Available Scenarios". Under "Existing Projects", there is a dropdown menu currently showing "test4". Under "Available Scenarios", there is a dropdown menu currently showing "scn_default". Below these, there are two date/time input fields. The first is labeled "Begin Date with format: yyyy/MM/dd:HH:mm:ss" and contains the text "1970/01/01:00:00:00". The second is labeled "End Date with format: yyyy/MM/dd:HH:mm:ss" and contains the text "2015/11/26:16:11:49". At the bottom of the dialog, there are two buttons: "Open" and "Cancel".

There are 2 combobox, the first for the projects (which is set with the most recent created project), the second are the scenarios of the set project (set also with the most recent scenario).

In this form, you can also modify the begin and end date of the scenario before opening the scenario, the new dates are saved.

3.3 Adding or modifying a scenario in an opened project

After clicking File → Change/Create Scenario, you get a Dialog as shown below :

LogFouineur : Create a New Project

Add or change scenario for project : test4

Existing Scenario scn_default

Create a new Scenario

Begin Date
yyyy/MM/dd:HH:mm:ss

End Date
yyyy/MM/dd:HH:mm:ss

- You can modify the date of the current scenario.
- Or you can create a new scenario with filling the TextField of the new scenario, and filling the Begin and End date in respect with the format.