AspectsPerf

MONITORING JAVA APPLICATIONS WITH ASPECTJ

Date	*Version*	*Modification*	*Author*
2012/12/31	1.0.0	Initialization	Jean-Louis PASTUREL

Table des matières

1 INTRODUCTION

1.1 A rapid presentation of AspectJ

1.1.1 URLs to visit :

http://www.eclipse.org/aspectj/

«I want my AOP» by Ramnivas Laddad en 3 parties (Author of **AspectJ in Action**)

1.1.2 joinpoint

It is a particular point of code that can be reached by an aspect.

The principal **joinpoints** handled by AspectJ are :

- method call
- method execution
- constructor call
- constructor execution
- field get
- field set
- pre-initialization
- initialization
- static initialization
- handler
- advice execution

1.1.3 pointcut

A pointcut is an association of joinpoints defined by a set of key words, boolean operators (AND OR NOT) , and regular expressions. There are also joker like ***** and **..**

The documentation downloaded with the AspectJ framework, gives a more complete definition of these concepts.

Example of a definition of a **pointcut** :

pointcut myPointCut(): within(jlp..*) AND call(public * jlp..*(..)) AND ! cflowbelow(call(public * jlp..*(..)))

This pointcut captures all the public methods of the classes of packages jlp.* that return anything and have 0 or n arguments. In addition, the methods that are in the flow of the latter are excluded from the scope of this pointcut (!**cflowbelow(call(public * jlp..*(..)))**).

1.1.4 advice

An advice is a piece of code that is executed when a pointcut is reached.

There are 3 principal type of advice (see the official documentation to learn more about other advices) :

- **Before**
- **After**
- **Around**

1.2 Type of weaving

With AspectJ, you can realize two type of weaving :

- at compilation Time (CTW) from source or binary java bytecode
- at runtime (Load Time Weaving => LTW)

The first mode of weaving(CTW) affects permanently the behavior of the application. The second mode does not affect the application code. With LTW, when you deactivate the javaagent, the application retrieves the initial behaviour.

It is the second mode (**LTW)** that is used in **aspectsPerf**.

1.3 Presentation of aspectsPerf

AspectsPerf is an application consisting of a set of AspectJ and Java classes permitting to log or expose JMX Mbeans to follow the behavior of a java application without modifying the code of the application.

Aspects may be used in different domains :

- Counting instantiation of Objects
- tracking particulars methods duration when a Profiler can't be used (total time and CPU time)
- tracking duration of Jolt services, web services, SQL requests ...
- exposing attributes of classes that are not JMX Mbean, or not registered (pool Apache DBCP, unregistered C3P0 pool)
- tracking size of http session or JPA session

…

This tool comes with a set of Aspects

AspectsPerf may be enriched by other Aspects when new needs arise.

1.4 Licenses

AspectsPerf is an application developed in Java and AspectJ and uses several Java API:

AspectsPerf Core : Apache 2 License http://www.apache.org/licenses/LICENSE-2.0.html

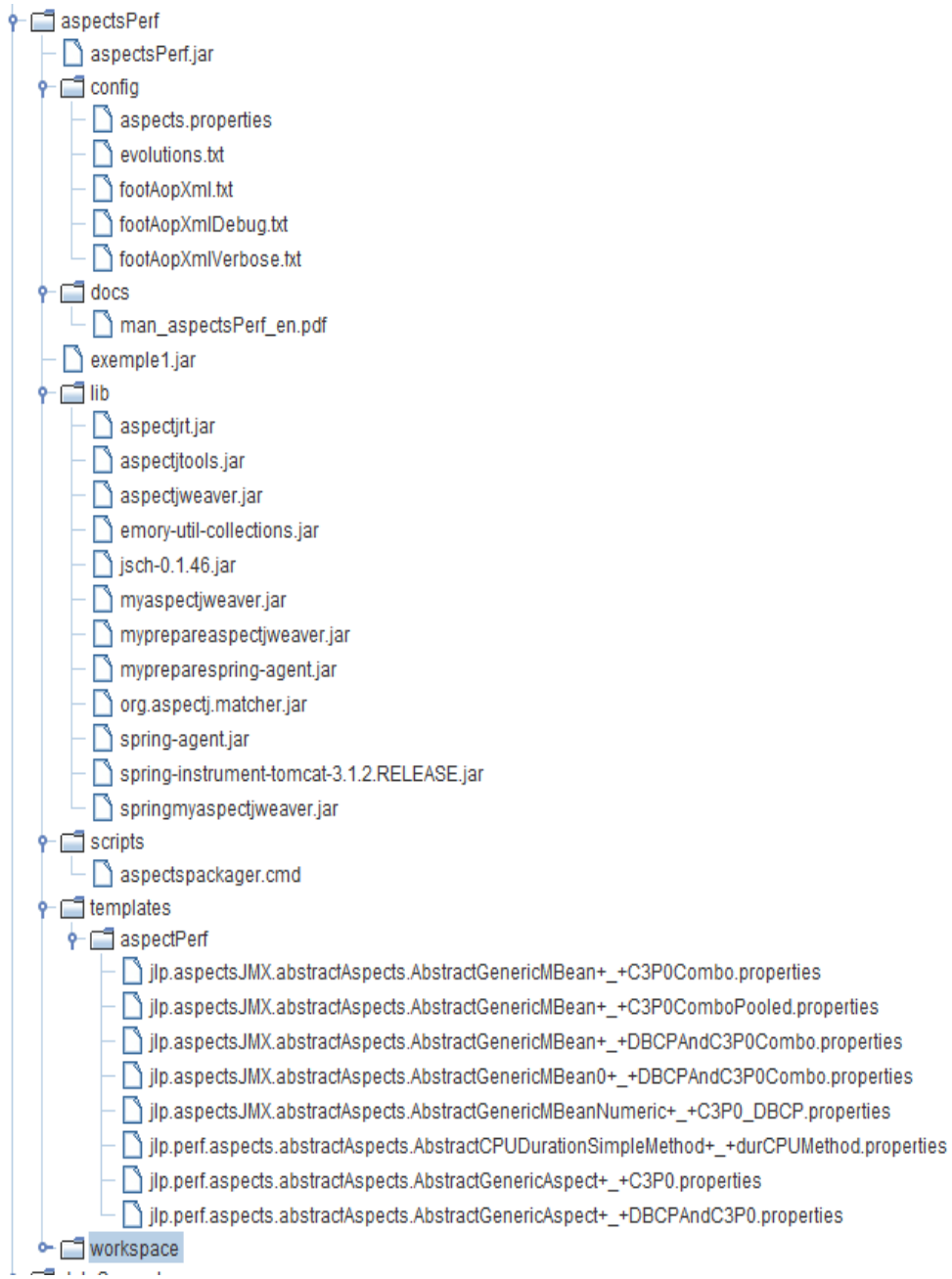
Jsch (BSD License) : http://www.jcraft.com/jsch/LICENSE.txt

AspectJ (Eclipse Project) : Common Public License http://eclipse.org/legal/cpl-v10.html

2 AspectsPerf

2.1 Installation of aspectsPerf

The application is zip or a gz package. Copy the archive in the target directory and unpack it. The directories are created as shown in the screen-shot below.



The workspace folder must be created by hand.

2.2 Configure <aspectsPerf_HOME>/scripts/aspectpackager.cmd

```
SET root=D:\opt\aspectsPerf
SET workspace=D:\opt\aspectsPerf\workspace
SET CLASSPATH=%root%\aspectsPerf.jar
SET CLASSPATH=%CLASSPATH%;%root%\lib\aspectjweaver.jar

java  -Xms128M -Xmx128M -Droot=%root% -Dworkspace=%workspace% -classpath
%CLASSPATH% jlp.perf.aspects.gui.AspectsPackager
```

The environment variables **root** and **workspace** must be correctly according to your installation.

If java is not in your path, replace **java** by the full path. Be carefull with " =>

"C:\Program Files\Java\<your_jre>\bin\java"

2.3 Test of the installation and QuickStart

The jar archive **aspectsPerf.jar** contains all the Aspects and some class to test the packaging.

The class that we will use to test the installation is :

`jlp.aspectj.test.TestingClass`

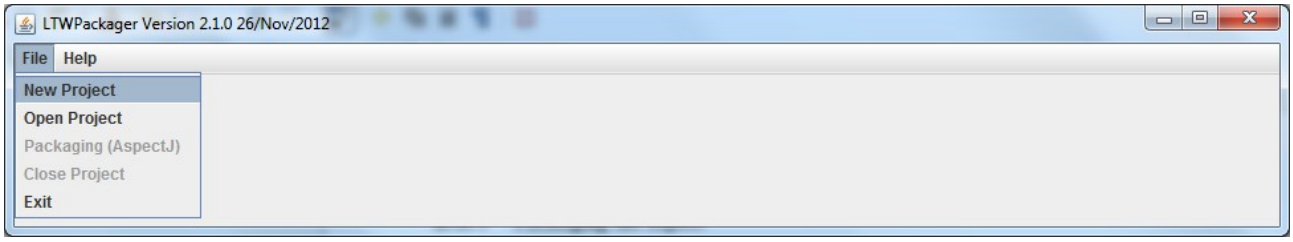
This class extends Runnable and has several methods. The aspect :

`jlp.perf.aspects.abstractAspects.AbstractDurationSimpleMethod`

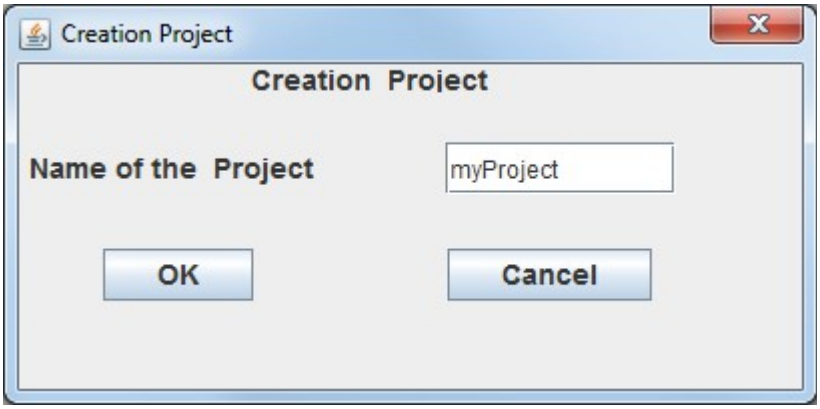
permits to follow the duration of the methods .

2.3.1 Packaging the aspect

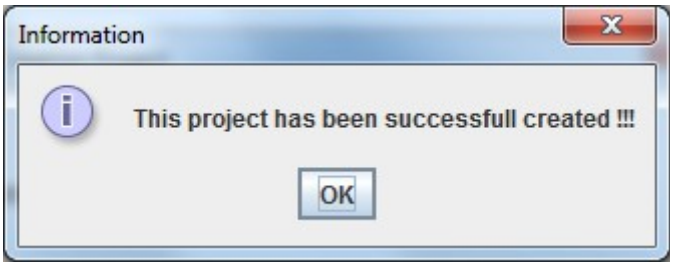
Launch the application with the updated script : **aspectpackager.cmd**



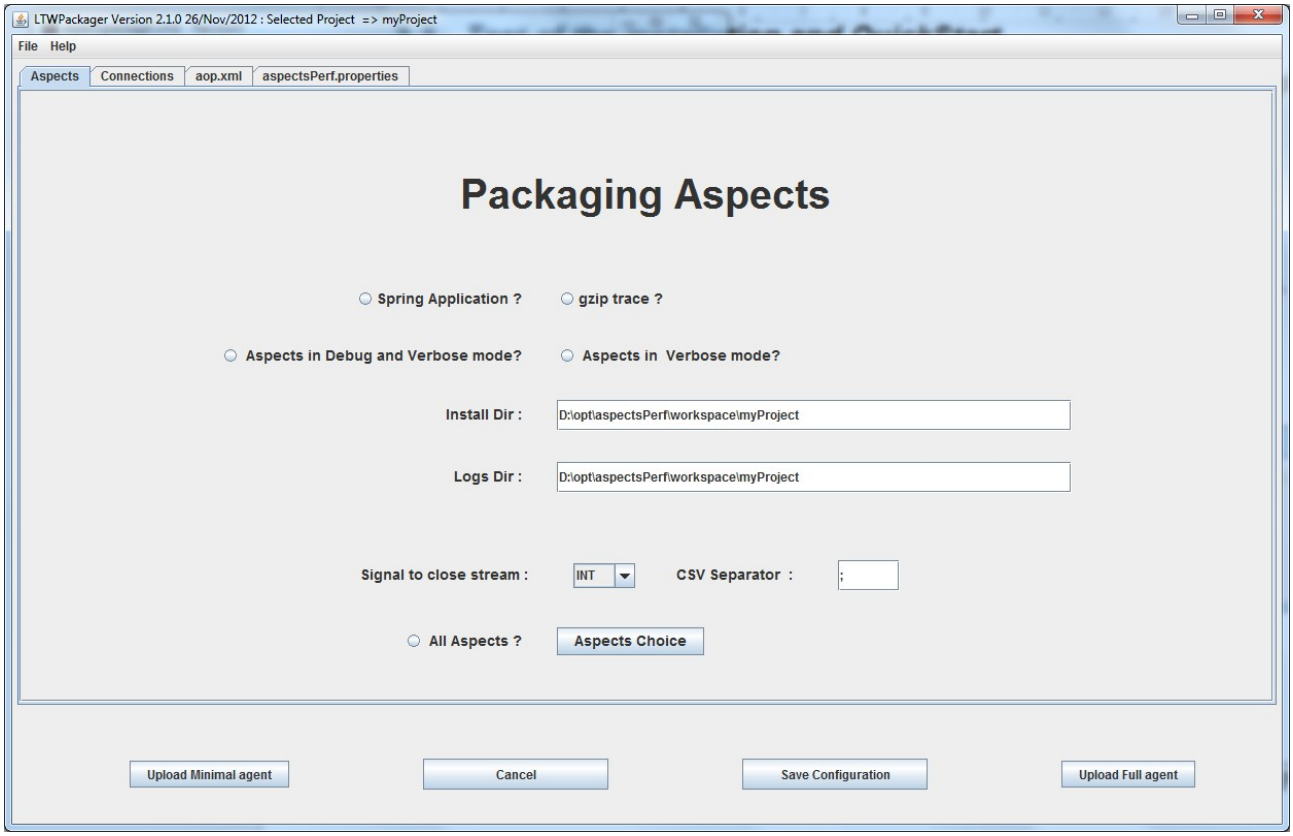
First thing to do is creating a new project =>



and OK,



OK again, the screen below appears :



There are some fields and radioButton to set.

The most important are :

- when aspect logs traces in a file, you can write directly in gzip file, to retrieve a correct gzip file you must stop the JVM, or send a signal (signal to close stream). Sending a signal works correctly on unix servers. The behavior on Windows (INT signal / CRTL C) is not guaranteed. If gzip is not selected, the file is a text file that is can be read directly.

- Logs Dir specifies the location of the traces if any, on the target server.

- Install Dir is used when you upload the javaagent to the target server.

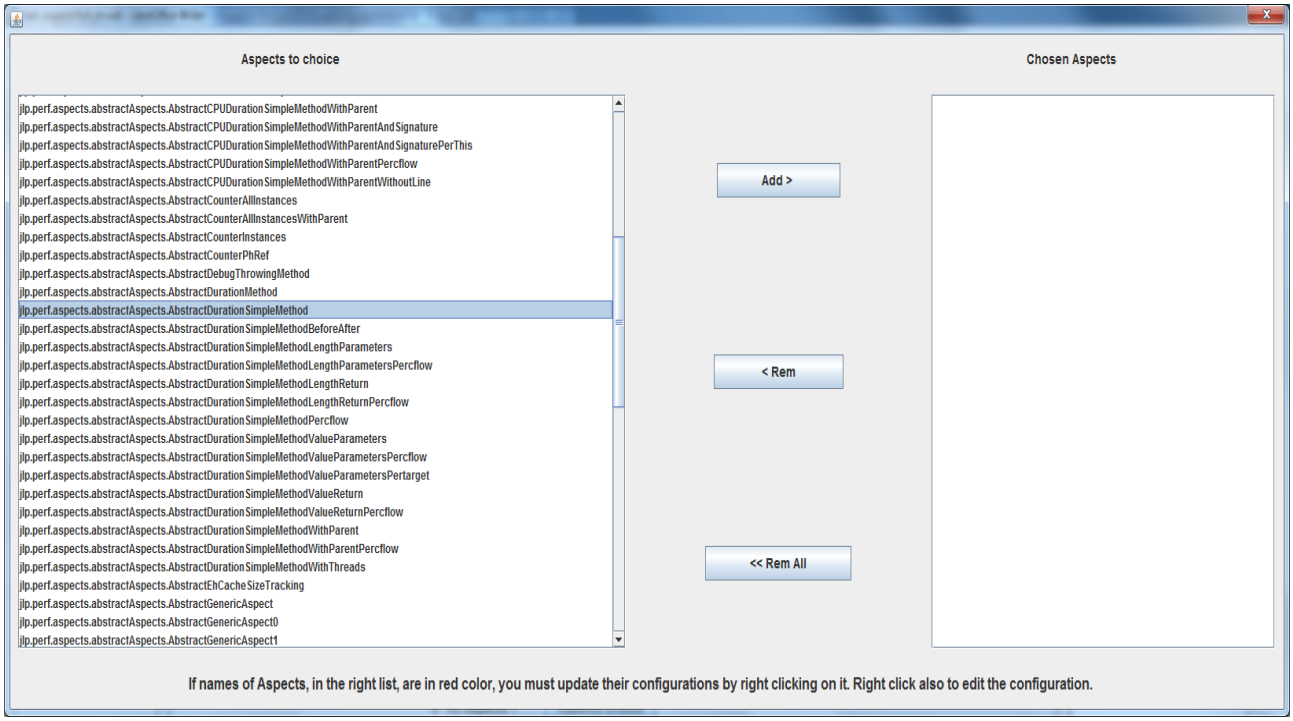
- CSV separator is generally ;

- Signal to close Stream : in general HUP for unix and INT for Windows

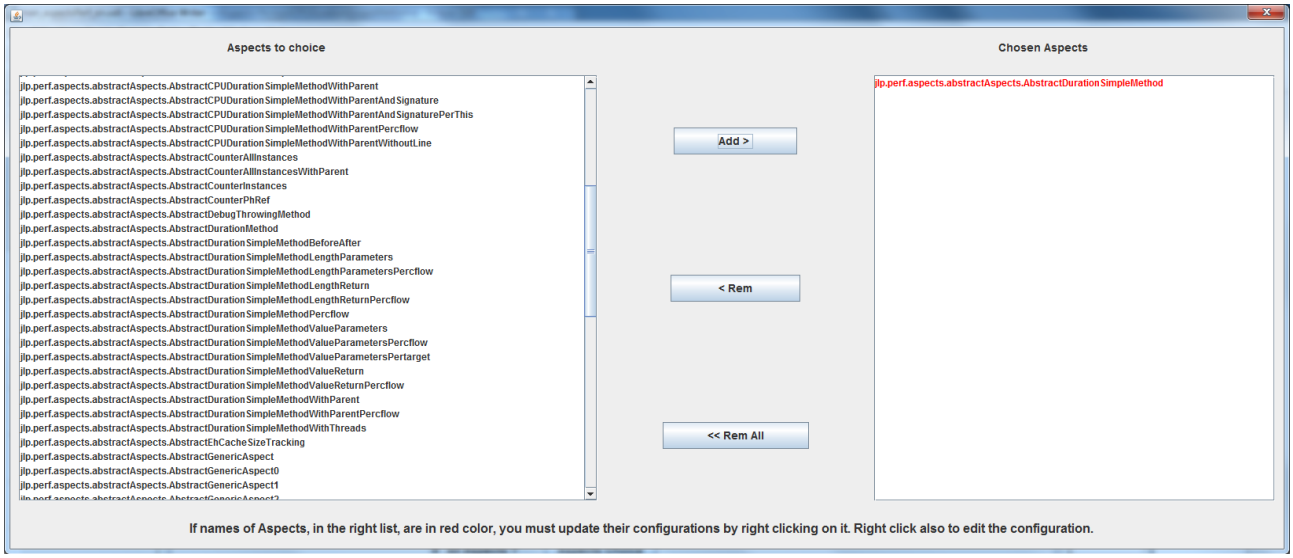
The tab **Connections** permits to upload the package **myaspectjweaver.jar** to the target servers (when clicking on “Upload Full agent” or “Upload Minimal agent”).

The tab **aop.xml** and **aspectsPerf.properties** contains the result of the selection. These tabe are read only.

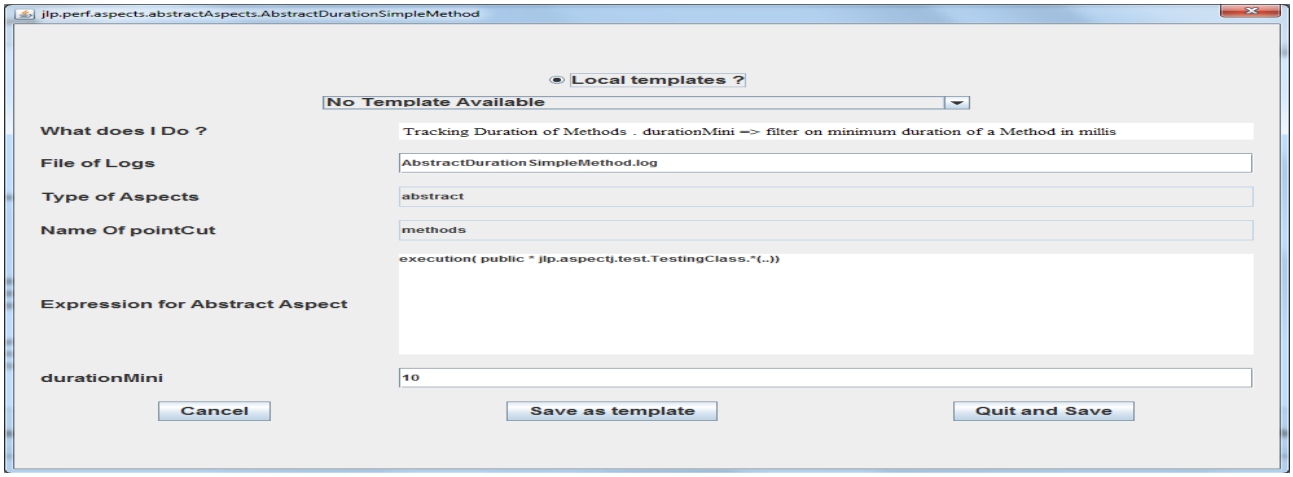
The next step is to choose an aspect in the list of available aspects : click on “**Aspect choice**”



At the left in the list of aspects, we choose
jlp.perf.abstractAspects.AbstractDurationSimpleMethod



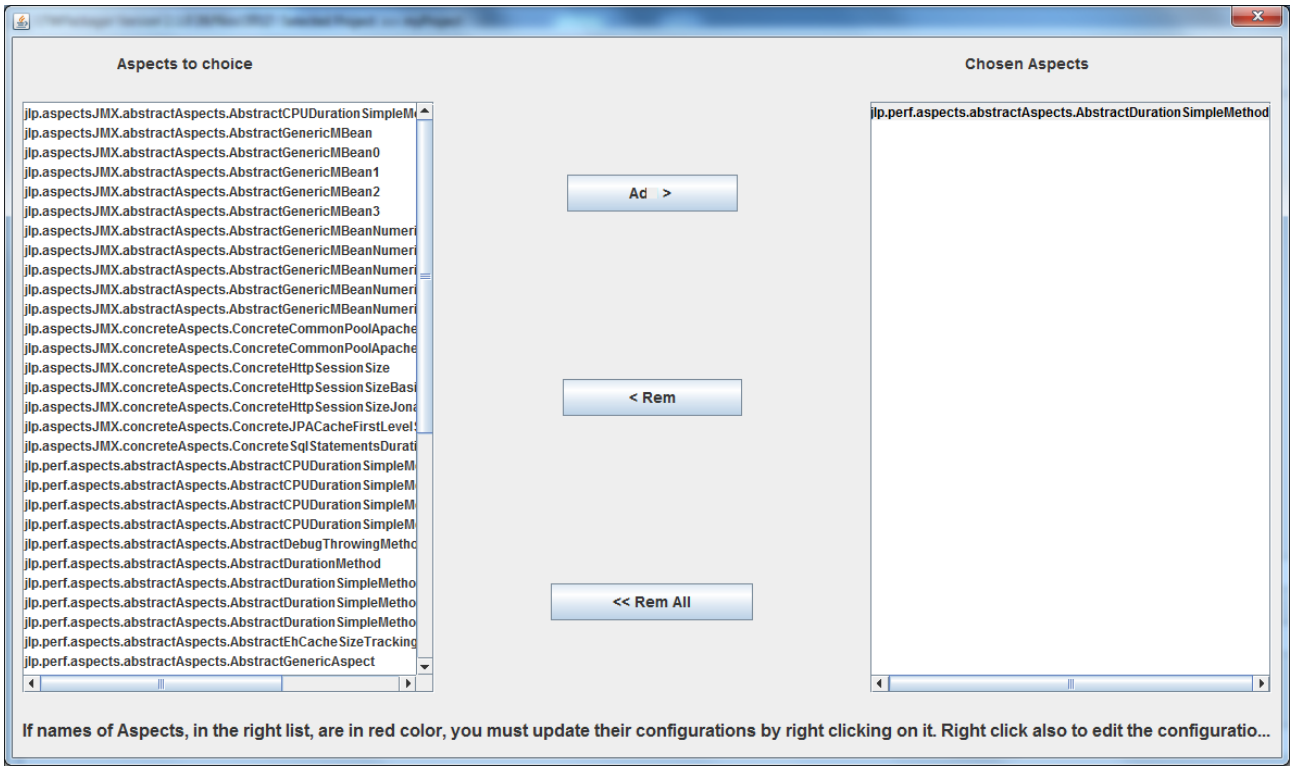
The chosen aspect appears in red colors, at the right side. You must right click to configure it.



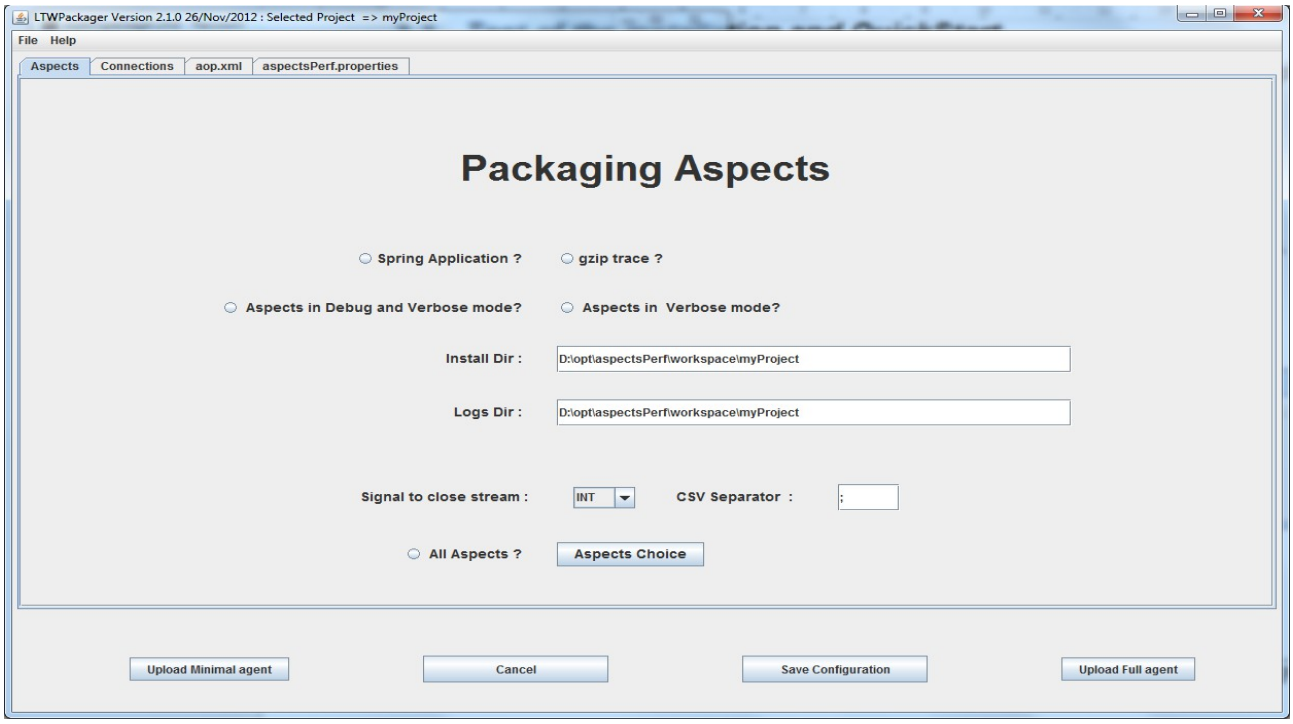
This is the screen to configure the aspect. The most important is to set a correct **pointcut** in the expression TextArea (see aspectJ documentation for more details)

You can click on Quit and Save.

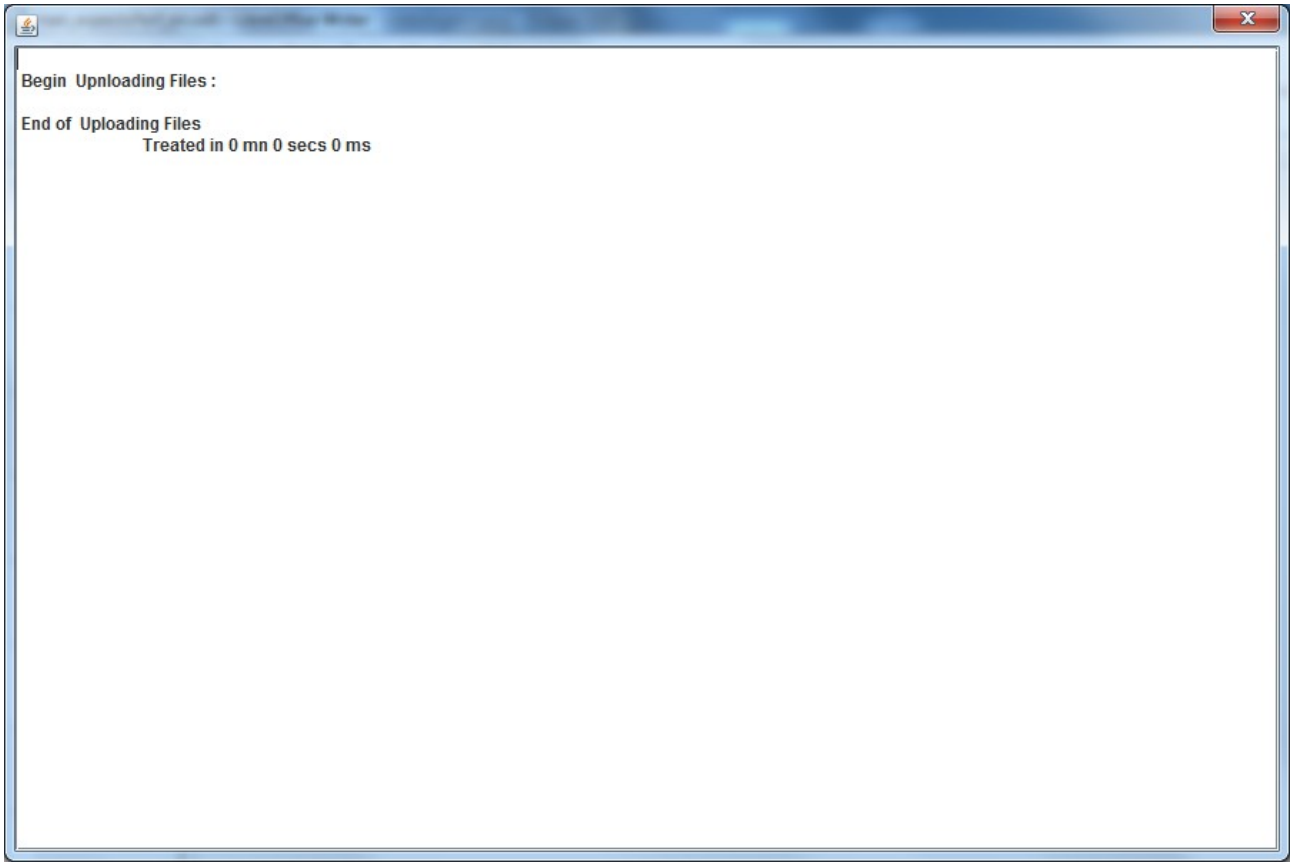
The button **Save as Template** will be explained feather in the document.



The aspect appears in black color, and you can close the window.



Click on **Save Configuration** and **Upload Minimal agent**.



When this dialog is closed(), there is a jar file , in the directory myProject:

myaspectjweaver.jar that is the java agent that we will use against our TestingClass.

2.3.2 Testing the aspect

Launch the application with the

To test the agent, go to the directory “myProject” :

```
cd D:\opt\aspectsPerf\workspace\myProject

D:\opt\aspectsPerf\workspace\myProject>java -javaagent:.\myaspectjweaver.jar
jlp.aspectj.test.TestingClass 20 1000000
```

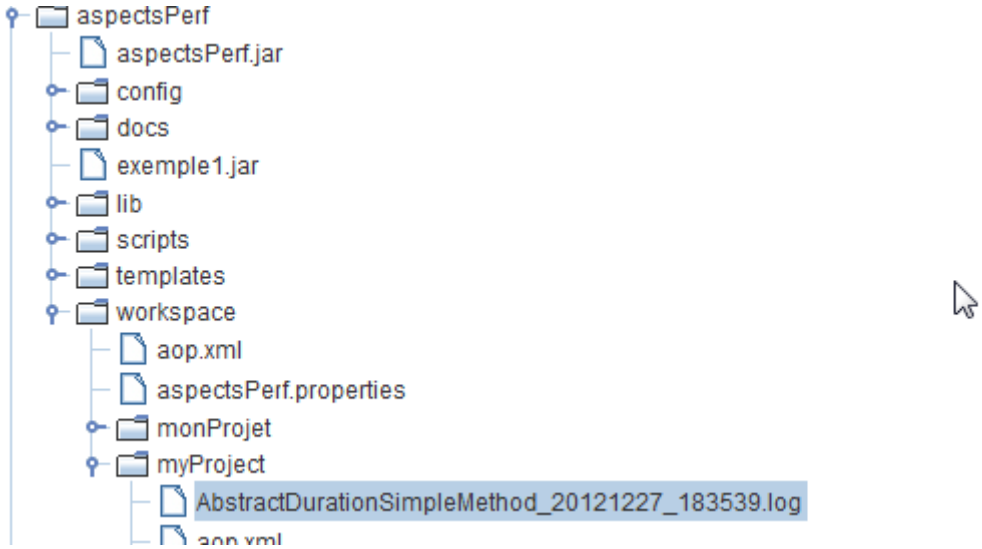
You get the output like this :

```
C:\Windows\system32\cmd.exe
Chargement fichier properties
Chargement fichier properties dans le jar
AbstractDurationSimpleMethod14 fichier trace = D:\opt\aspectsPerf\workspace\myPr
oject\AbstractDurationSimpleMethod.log
[Instrumentation Aspects by AspectsPerf ] Deb Creation Trace os.name = WINDOWS 7

AspectsPerf Version : 2.1.0 of 21/Dec/2012. Author JL Pasturel
[Instrumentation Aspects by AspectsPerf ] jlp.perf.aspects.abstractAspects.Trace
 nomFileLogs =  D:\opt\aspectsPerf\workspace\myProject\AbstractDurationSimpleMet
hod_20121227_183539.log
[Instrumentation Aspects by AspectsPerf ] jlp.perf.aspects.abstractAspects.Trace
 title =  ####time;Class.methods;time in millisecondes

sh n est pas null
Before handle signal
After handle signal
[Instrumentation Aspects by AspectsPerf ] Signal Handler initialized by signal :
 [INT]
thread_2.myShortMethod : I will sleep for 100 ms
thread_3.myShortMethod : I will sleep for 100 ms
thread_7.myShortMethod : I will sleep for 100 ms
thread_4.myShortMethod : I will sleep for 100 ms
thread_0.myShortMethod : I will sleep for 100 ms
thread_1.myShortMethod : I will sleep for 100 ms
thread_13.myShortMethod : I will sleep for 100 ms
thread_12.myShortMethod : I will sleep for 100 ms
thread_11.myShortMethod : I will sleep for 100 ms
thread_10.myShortMethod : I will sleep for 100 ms
thread_9.myShortMethod : I will sleep for 100 ms
thread_8.myShortMethod : I will sleep for 100 ms
thread_6.myShortMethod : I will sleep for 100 ms
thread_5.myShortMethod : I will sleep for 100 ms
thread_19.myShortMethod : I will sleep for 100 ms
thread_18.myShortMethod : I will sleep for 100 ms
thread_17.myShortMethod : I will sleep for 100 ms
thread_16.myShortMethod : I will sleep for 100 ms
thread_15.myShortMethod : I will sleep for 100 ms
thread_14.myShortMethod : I will sleep for 100 ms
thread_5.myStressingMethod :  limit=1000000
thread_15.myStressingMethod :  limit=1000000
thread_0.myStressingMethod :  limit=1000000
thread_2.myStressingMethod :  limit=1000000
thread_12.myStressingMethod :  limit=1000000
thread_11.myStressingMethod :  limit=1000000
thread_10.myStressingMethod :  limit=1000000
thread_19.myStressingMethod :  limit=1000000
thread_4.myStressingMethod :  limit=1000000
thread_3.myStressingMethod :  limit=1000000
thread_18.myStressingMethod :  limit=1000000
```

To stop, type a CRTL C and N, the application stop,

```
C:\Windows\system32\cmd.exe
thread_3.myStressingMethod :  limit=1000000
thread_9.myShortMethod : I will sleep for 100 ms
thread_7.myLongMethod : I will sleep for 1s
thread_8.myShortMethod : I will sleep for 100 ms
thread_11.myShortMethod : I will sleep for 100 ms
thread_15.myLongMethod : I will sleep for 1s
thread_4.myStressingMethod :  limit=1000000
thread_6.myShortMethod : I will sleep for 100 ms
thread_5.myLongMethod : I will sleep for 1s
thread_0.myShortMethod : I will sleep for 100 ms
thread_18.myShortMethod : I will sleep for 100 ms
thread_14.myLongMethod : I will sleep for 1s
thread_3.myLongMethod : I will sleep for 1s
thread_1.myStressingMethod :  limit=1000000
thread_4.myLongMethod : I will sleep for 1s
thread_11.myStressingMethod :  limit=1000000
thread_8.myStressingMethod :  limit=1000000
thread_9.myStressingMethod :  limit=1000000
thread_1.myLongMethod : I will sleep for 1s
thread_6.myStressingMethod :  limit=1000000
thread_13.myShortMethod : I will sleep for 100 ms
thread_17.myShortMethod : I will sleep for 100 ms
thread_18.myStressingMethod :  limit=1000000
thread_2.myShortMethod : I will sleep for 100 ms
thread_6.myLongMethod : I will sleep for 1s
thread_0.myStressingMethod :  limit=1000000
thread_19.myShortMethod : I will sleep for 100 ms
thread_8.myLongMethod : I will sleep for 1s
thread_9.myLongMethod : I will sleep for 1s
thread_11.myLongMethod : I will sleep for 1s
thread_18.myLongMethod : I will sleep for 1s
thread_0.myLongMethod : I will sleep for 1s
AspectsPerf Version : 2.1.0 of 21/Dec/2012. Author JL Pasturel
[Instrumentation Aspects par AspectsPerf ] Received signal: SIGINT
AspectsPerf Version : 2.1.0 of 21/Dec/2012. Author JL Pasturel
[Instrumentation Aspects by AspectsPerf ] Closing all streams and re-opening new
 OutPutStreams
[Instrumentation Aspects by AspectsPerf ]Continue (O/N or Y/N) : ?
N
AspectsPerf Version : 2.1.0 of 21/Dec/2012. Author JL Pasturel
[Instrumentation Aspects by AspectsPerf ] Received a control C
[Instrumentation Aspects by AspectsPerf ] Closing all OutpuStreams

D:\opt\aspectsPerf\workspace\myProject>_
```

and you get a log trace :



This file is a csv file that can be loaded and charted in **MS Excel**, or **LibreOffice/OpenOffice**. It can be also graphed in a tool (I am the author) like **swingScaViewer** :https://github.com/PASTJL/swingScaViewer

```
####time;Class.methods;time in millisecondes(ms)

2012/12/31 10:04:42.805;jlp.aspectj.test.TestingClass.myShortMethod_line_41;99.068
2012/12/31 10:04:42.805;jlp.aspectj.test.TestingClass.myShortMethod_line_41;98.906
2012/12/31 10:04:42.805;jlp.aspectj.test.TestingClass.myShortMethod_line_41;98.899
2012/12/31 10:04:42.805;jlp.aspectj.test.TestingClass.myShortMethod_line_41;98.971
2012/12/31 10:04:42.805;jlp.aspectj.test.TestingClass.myShortMethod_line_41;96.503
2012/12/31 10:04:42.805;jlp.aspectj.test.TestingClass.myShortMethod_line_41;99.177
2012/12/31 10:04:42.821;jlp.aspectj.test.TestingClass.myShortMethod_line_41;124.890
2012/12/31 10:04:42.821;jlp.aspectj.test.TestingClass.myShortMethod_line_41;125.313
2012/12/31 10:04:42.821;jlp.aspectj.test.TestingClass.myShortMethod_line_41;126.127
2012/12/31 10:04:42.821;jlp.aspectj.test.TestingClass.myShortMethod_line_41;127.247
2012/12/31 10:04:42.821;jlp.aspectj.test.TestingClass.myShortMethod_line_41;127.378
2012/12/31 10:04:42.821;jlp.aspectj.test.TestingClass.myShortMethod_line_41;128.493
2012/12/31 10:04:42.821;jlp.aspectj.test.TestingClass.myShortMethod_line_41;128.374
2012/12/31 10:04:42.821;jlp.aspectj.test.TestingClass.myShortMethod_line_41;124.893
2012/12/31 10:04:42.821;jlp.aspectj.test.TestingClass.myShortMethod_line_41;125.738
2012/12/31 10:04:42.836;jlp.aspectj.test.TestingClass.myShortMethod_line_41;127.007
2012/12/31 10:04:42.836;jlp.aspectj.test.TestingClass.myShortMethod_line_41;130.093
2012/12/31 10:04:42.836;jlp.aspectj.test.TestingClass.myShortMethod_line_41;137.182
2012/12/31 10:04:42.836;jlp.aspectj.test.TestingClass.myShortMethod_line_41;138.458
2012/12/31 10:04:42.836;jlp.aspectj.test.TestingClass.myShortMethod_line_41;140.142
2012/12/31 10:04:43.024;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;190.670
2012/12/31 10:04:43.086;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;291.751
2012/12/31 10:04:43.148;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;326.854
2012/12/31 10:04:43.148;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;355.697
2012/12/31 10:04:43.164;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;330.045
2012/12/31 10:04:43.195;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;359.560
2012/12/31 10:04:43.211;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;387.009
2012/12/31 10:04:43.226;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;385.950
2012/12/31 10:04:43.258;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;462.562
2012/12/31 10:04:43.273;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;447.320
2012/12/31 10:04:43.289;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;487.174
```

```
2012/12/31 10:04:43.320;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;499.061
2012/12/31 10:04:43.336;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;490.445
2012/12/31 10:04:43.336;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;507.808
2012/12/31 10:04:43.336;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;538.359
2012/12/31 10:04:43.351;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;544.049
2012/12/31 10:04:43.351;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;517.899
2012/12/31 10:04:43.367;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;546.308
2012/12/31 10:04:43.382;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;553.959
2012/12/31 10:04:43.398;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;566.224
2012/12/31 10:04:44.053;jlp.aspectj.test.TestingClass.myLongMethod_line_51;1014.040
2012/12/31 10:04:44.100;jlp.aspectj.test.TestingClass.myLongMethod_line_51;1000.233
2012/12/31 10:04:44.162;jlp.aspectj.test.TestingClass.myShortMethod_line_41;109.039
2012/12/31 10:04:44.209;jlp.aspectj.test.TestingClass.myShortMethod_line_41;108.987
2012/12/31 10:04:44.209;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;61.253
2012/12/31 10:04:44.240;jlp.aspectj.test.TestingClass.myLongMethod_line_51;1076.815
2012/12/31 10:04:44.240;jlp.aspectj.test.TestingClass.myLongMethod_line_51;1006.381
2012/12/31 10:04:44.256;jlp.aspectj.test.TestingClass.myLongMethod_line_51;1032.554
2012/12/31 10:04:44.256;jlp.aspectj.test.TestingClass.myLongMethod_line_51;1092.308
2012/12/31 10:04:44.256;jlp.aspectj.test.TestingClass.myLongMethod_line_51;1060.254
2012/12/31 10:04:44.256;jlp.aspectj.test.TestingClass.myLongMethod_line_51;1089.917
2012/12/31 10:04:44.272;jlp.aspectj.test.TestingClass.myLongMethod_line_51;1000.917
2012/12/31 10:04:44.272;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;62.366
2012/12/31 10:04:44.287;jlp.aspectj.test.TestingClass.myLongMethod_line_51;1002.711
2012/12/31 10:04:44.303;jlp.aspectj.test.TestingClass.myLongMethod_line_51;1007.600
2012/12/31 10:04:44.350;jlp.aspectj.test.TestingClass.myShortMethod_line_41;108.987
2012/12/31 10:04:44.350;jlp.aspectj.test.TestingClass.myLongMethod_line_51;1014.087
2012/12/31 10:04:44.350;jlp.aspectj.test.TestingClass.myShortMethod_line_41;108.987
2012/12/31 10:04:44.365;jlp.aspectj.test.TestingClass.myShortMethod_line_41;108.963
2012/12/31 10:04:44.365;jlp.aspectj.test.TestingClass.myShortMethod_line_41;109.025
2012/12/31 10:04:44.365;jlp.aspectj.test.TestingClass.myShortMethod_line_41;109.441
2012/12/31 10:04:44.365;jlp.aspectj.test.TestingClass.myLongMethod_line_51;1023.116
2012/12/31 10:04:44.412;jlp.aspectj.test.TestingClass.myStressingMethod_line_63;65.688
```

3 User Guide

The principles are explained in the quick start above, now we are looking for more advanced uses.

The tool aspectjpackager is composed by a set of AspectJ. These aspectJ can be packaged in a single jar (myaspectjweaver.jar) that contains also the LTW agent of the project Eclipse/AspectJ.

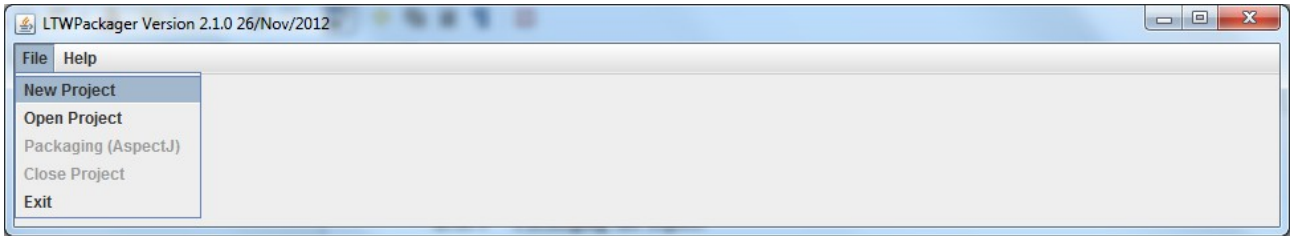
In the **META-INF directory** of the jar, there are the files :

- **aop.xml** (to weave the chosen aspects at LTW) ,
- **aspectsPerf.properties** (that contains customs parameters for each chosen aspect)
- **MANIFEST.MF** that configures the LTW java agent.

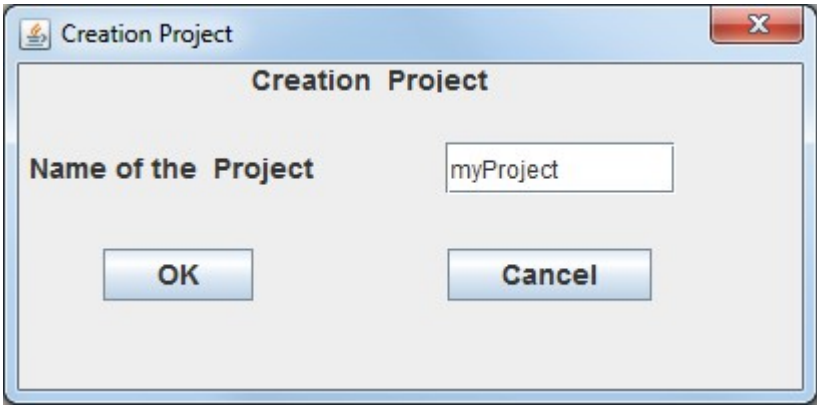
3.1 Starting the application

Explained above and copied almost as is below .

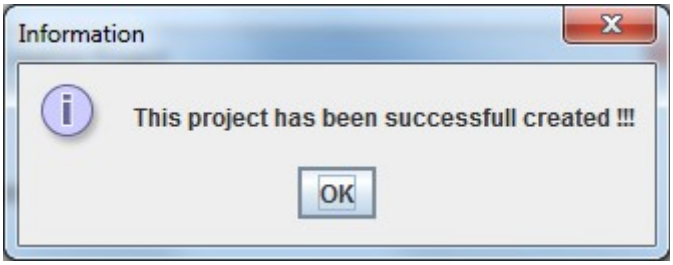
Launch the application with the updated script : **aspectpackager.cmd**



First thing to do is creating a new project =>

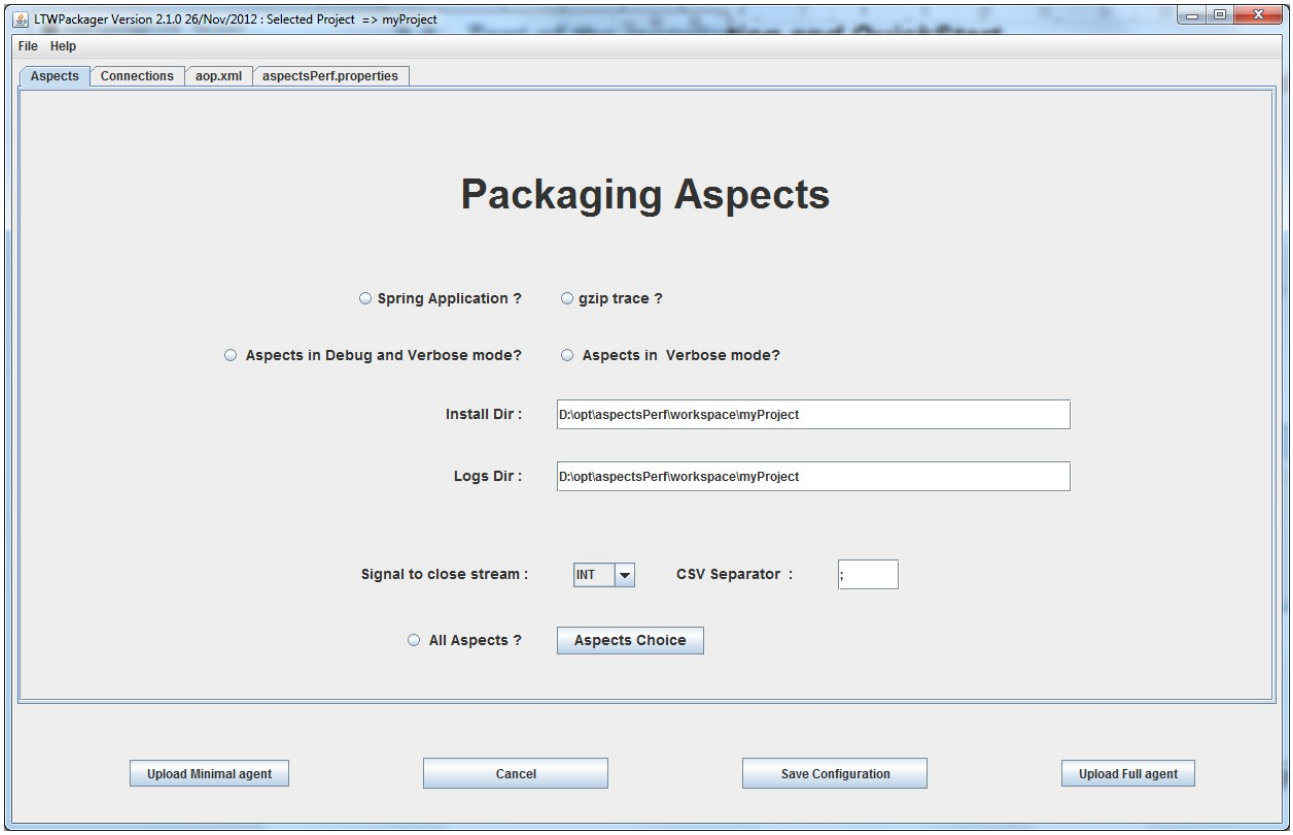


and OK,



The next time, you can directly open an existing project.

OK again, the screen below appears :



There are some fields and radio-buttons to set.

The most important are :

- when aspect logs traces in a file, you can write directly in gzip file, to retrieve a correct gzip file you must stop the JVM, or send a signal (signal to close stream). Sending a signal works correctly on unix servers. The behavior on Windows (INT signal / CRTL C) is not guaranteed. If gzip is not selected, the file is a text file that is can be read directly.

- Logs Dir specifies the location of the traces if any, on the target server.

- Install Dir is used when you upload the javaagent to the target server.

- CSV separator is generally “;”

- Signal to close Stream : in general HUP for unix and INT for Windows

- the radio Button All Aspects permits to chose in longer aspects list. But some aspects are redondant and not very useful. See further in the document, the presentation of the configuration file **aspects.properties** .

The tab **Connections** permits to upload the package **myaspectjweaver.jar** to the target servers (when clicking on “Upload Full agent” or “Upload Minimal agent”).

The tab **aop.xml** and **aspectsPerf.properties** contains the result of the selection. These two files are packaged in the jar **myaspectjweaver.jar** under the META-INF directory. These tabs are read only.

The radioduttons, debug and/or verbose mode permits to tune the aspects (to see if it weaves as expected).

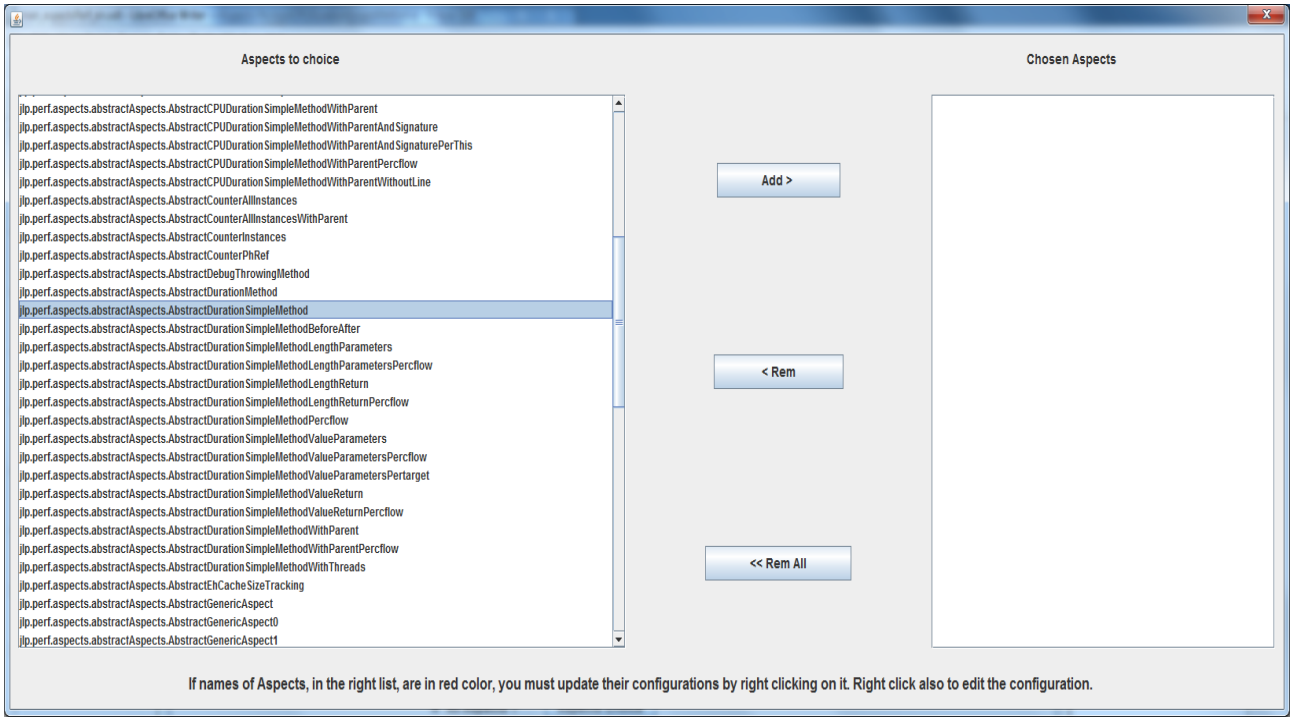
For Spring application, sometimes the LTW java agent that comes with AspectJ project doesn't run, and we have to use a specific Spring agent, that needs also a specific Spring main configuration. **aspectsPerf** configures the Spring agent in the jar myspringaspectjweaver.jar, but Spring may be also configured (see http://static.springsource.org/spring/docs/3.0.0.RC2/reference/html/ch07s08.html#aop-aj-ltw) . So this feature with a Spring application, is delivered with no guarantee.

The 4 buttons at the bottom of the screen were explained further in the user guide. Except for the button “Cancel”, there are used when the aspects are selected and configured

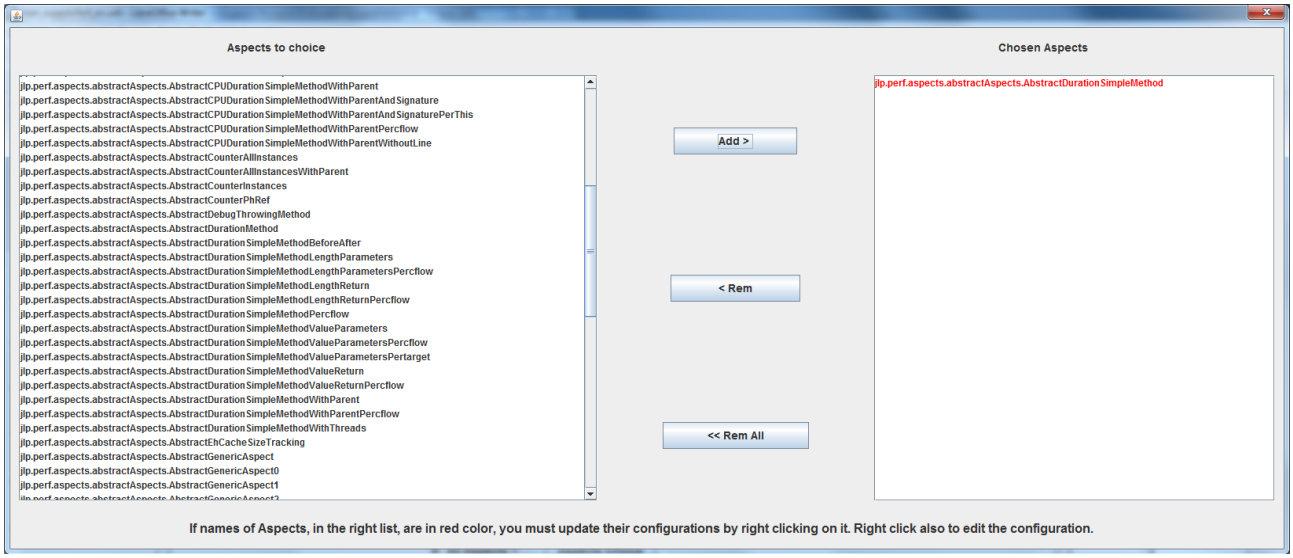
3.2 Choosing and configuring aspects

Be fore clicking on the button “**Aspects Choice**”, if you select the radio_button “**All Aspects**”, you can have access to a more numbers of Aspects, but certain are redundant or simply tries.

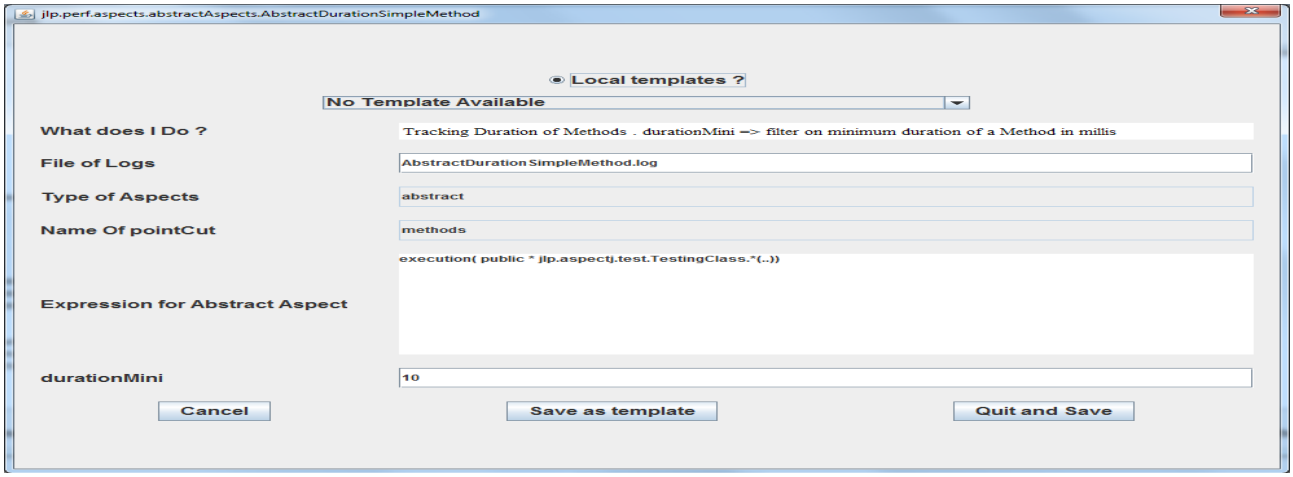
When “**Aspects Choice**” is clicked, the screen below appears :



At the left in the list of aspects, we choose
jlp.perf.abstractAspects.AbstractDurationSimpleMethod



The chosen aspect appears in red colors, at the right side. You must right click to configure it.



This is the screen to configure the aspect. The most important is to set a correct **pointcut** in the expression TextArea (see aspectJ documentation for more details)

For each Aspects, there are different JTextFields to fill, depending on number of parameters to set.

The different fields of this screen are :

- "**What does I do?**" explains in few words what the aspect does and how to fill the different fields. (read only)

- "**File of Logs"** : if any when aspects logs, the name of the file located in the log directory set in the first screen. (read/write)

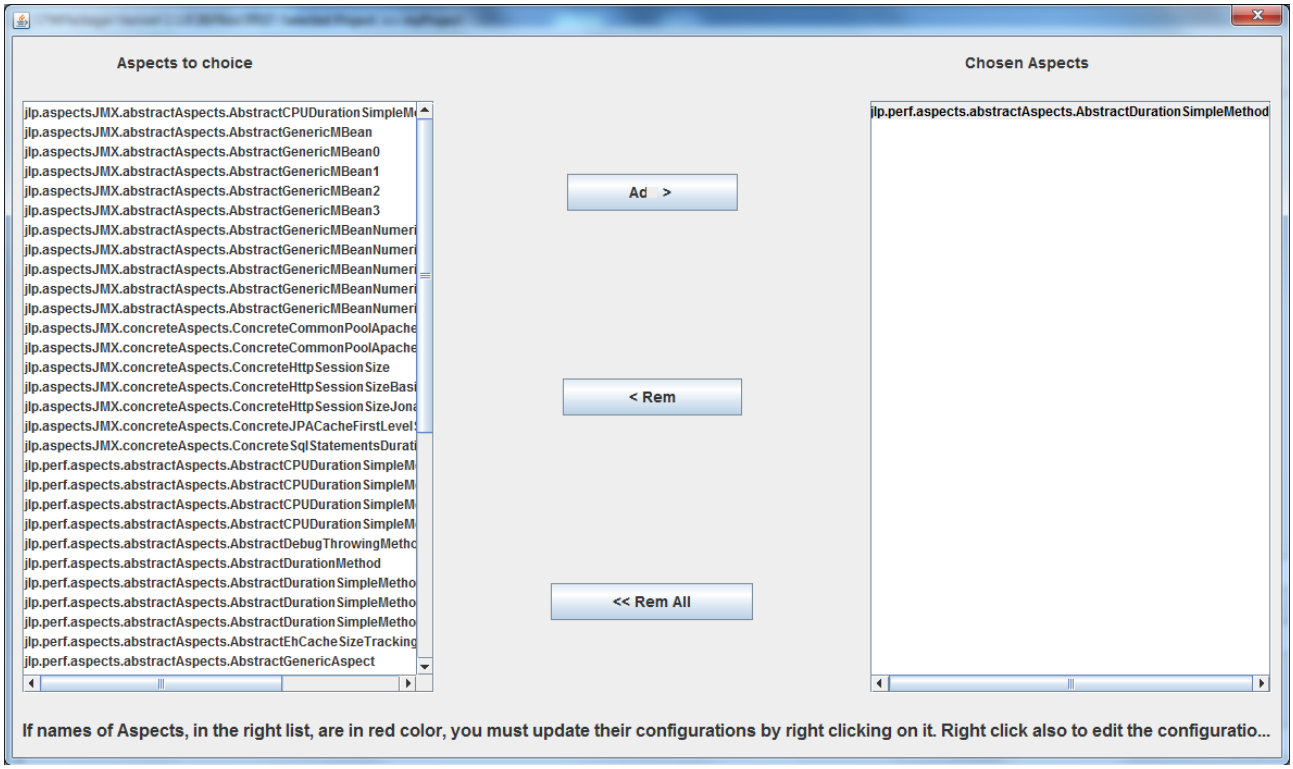
- "**Type of Aspects**" can be **abstract** or **concrete** (read only)

- "**Expression for Abstract Aspects**" if "**Type of Aspects**" is abstract, the expression of the pointcut must be filled carefully, and look at the stdout of the application, if the aspect is correct (no exception)

- "**durationMini**" is a specific parameter for the current aspect explained in the comment field above. It can be have others specifics fields depending of the aspect.

You can click on Quit and Save.

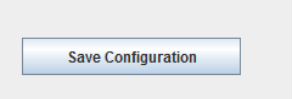
The button **Save as Template** will be explained feather in the document.



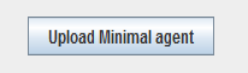
The aspect appears in black color, and you can close the window.

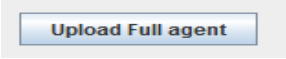
Afterward, you have to close this dialog box.

The last step is to save the configuration and to package the javaagent :

- Click on button 

- if you want a minimal javaagent (sufficient) with only the selected and used aspects (smaller)

click on 

if you want a full packaging click on 

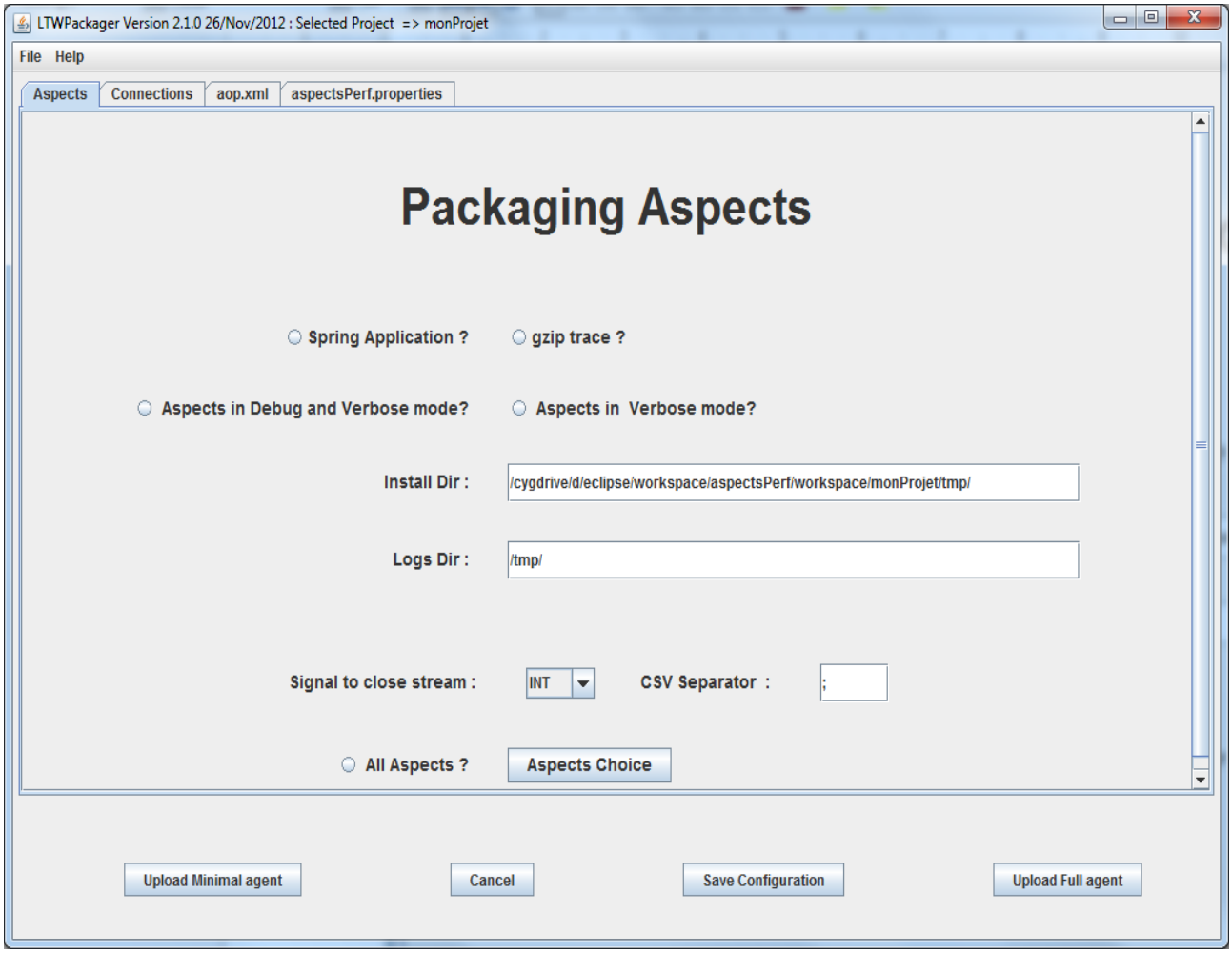
The first action, update the tabs **aop.xml** and **aspectsPerf.properties**

The last action, after having packaged the javaagent try to upload it to the destinations described in the tab **Connexions.** The javaagent is uploaded on the directory, of the target server, set in the textField : "**Install Dir** "

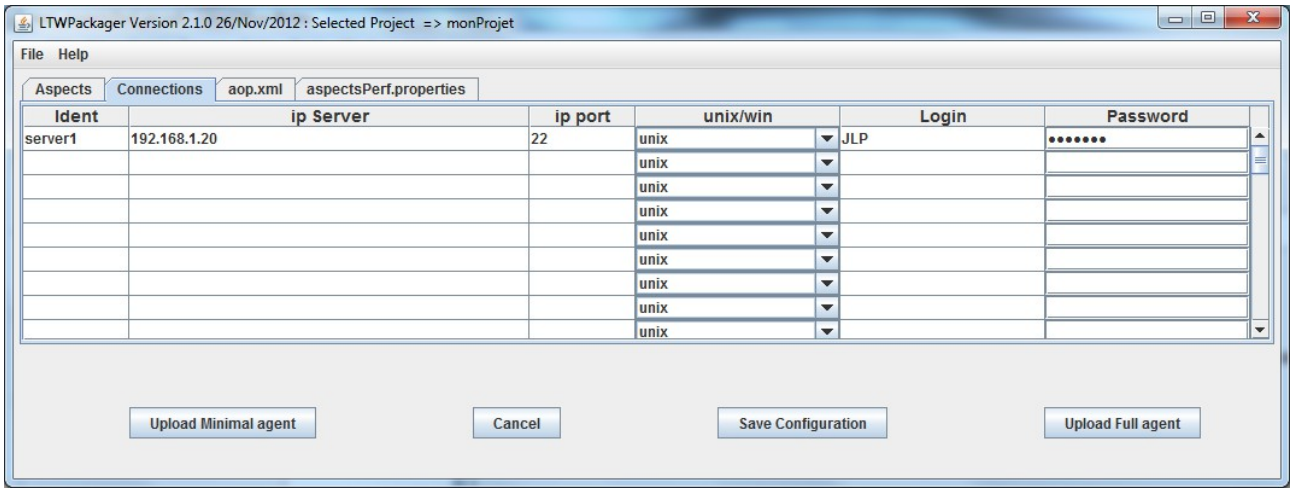
3.3 Tab Connections

Below a configuration, to upload to localhost (there is a service sshd running on my desktop with Cygwin). The target directory is :

/cygdrive/d/eclipse/workspace/aspectsPerf/workspace/monProjet/tmp/

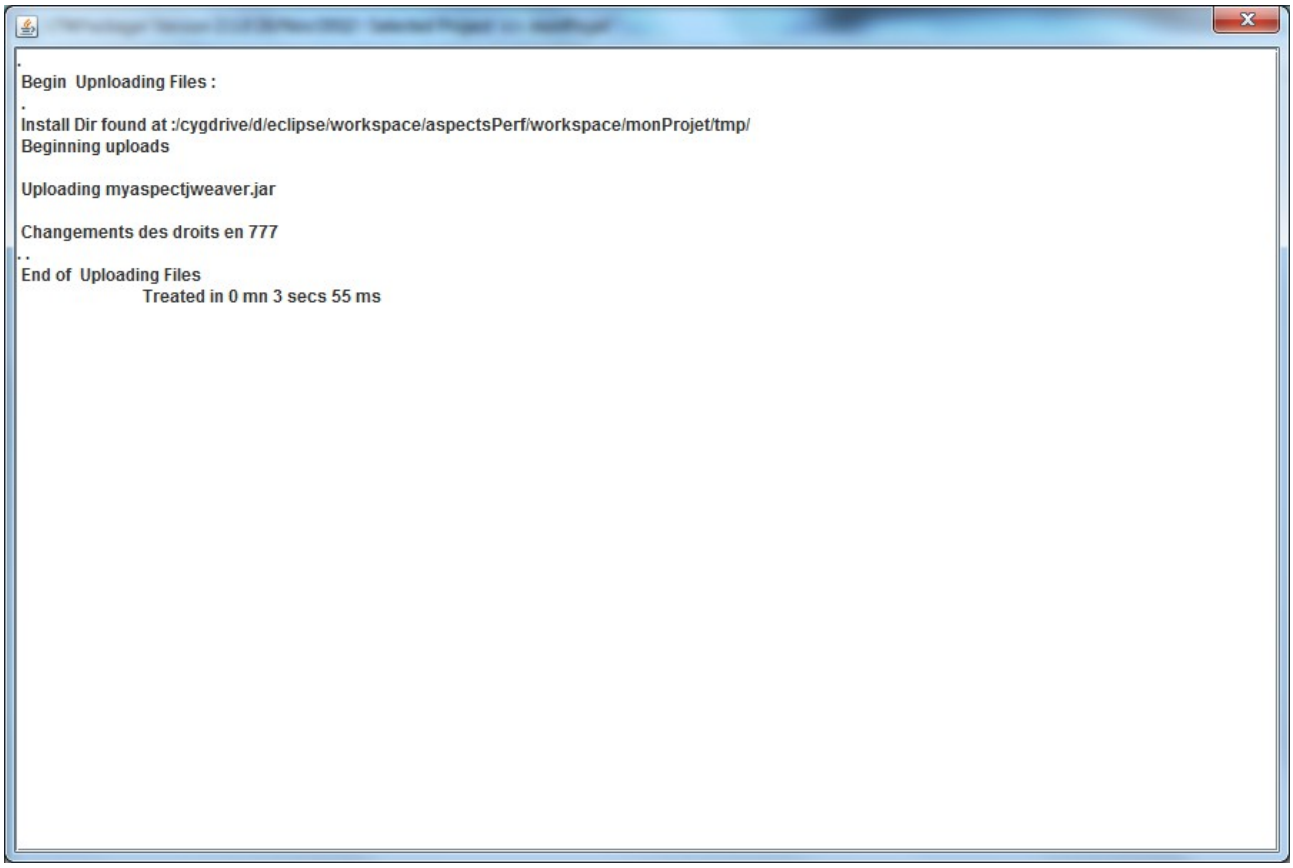


The tab **Connections** is filled as this :

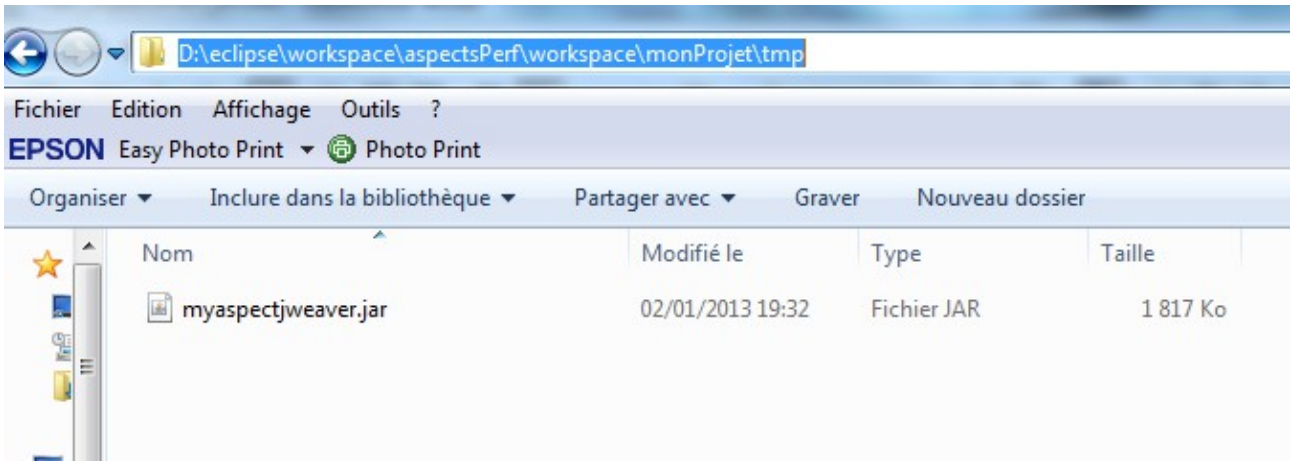


Cygwin ~ unix adapting the path for directories and files.

Clicking on Upload Minimal Agent =>



And file is uploaded :



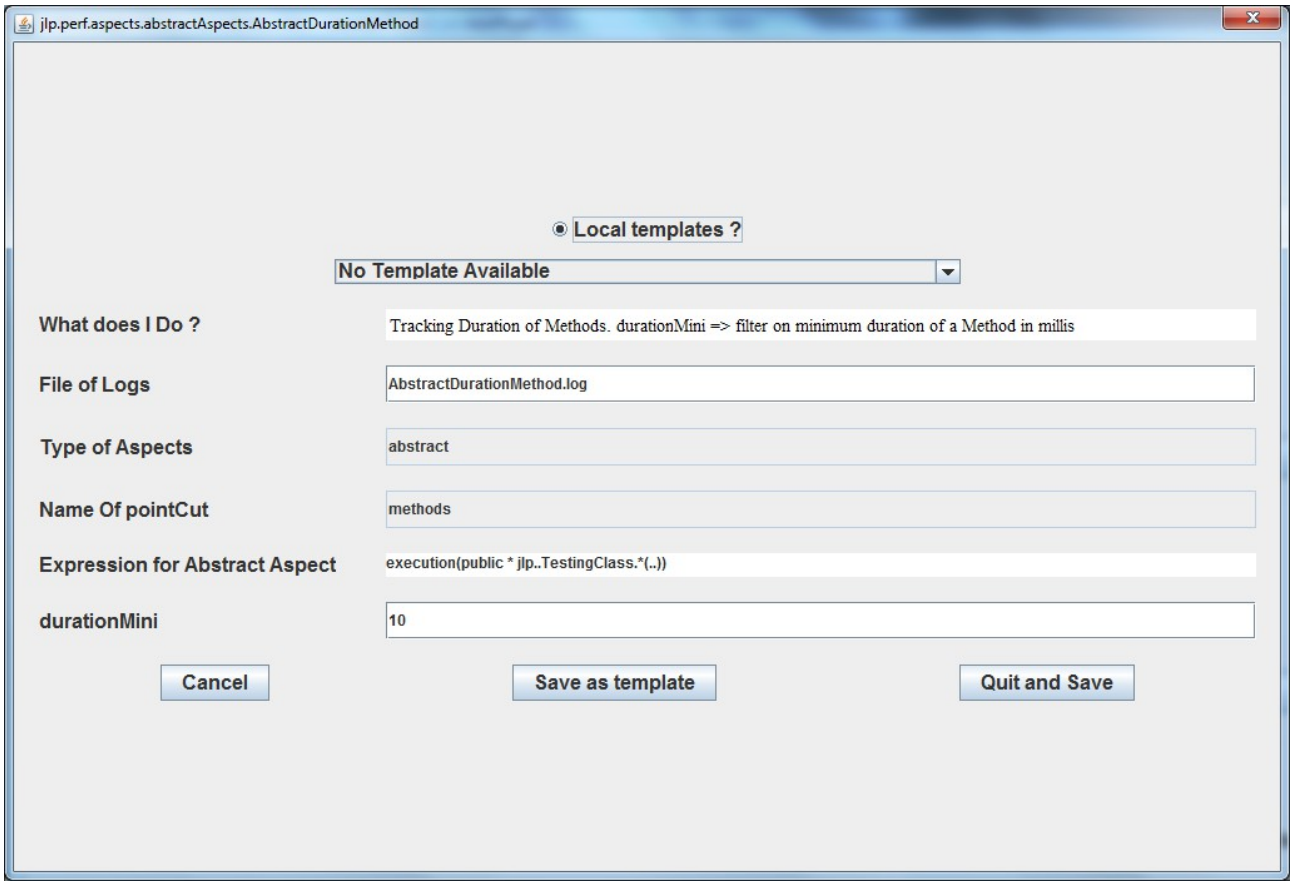
3.4 Dealing with templates

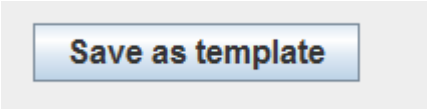
This feature permits to fill the configuration of the aspects when a template exists.

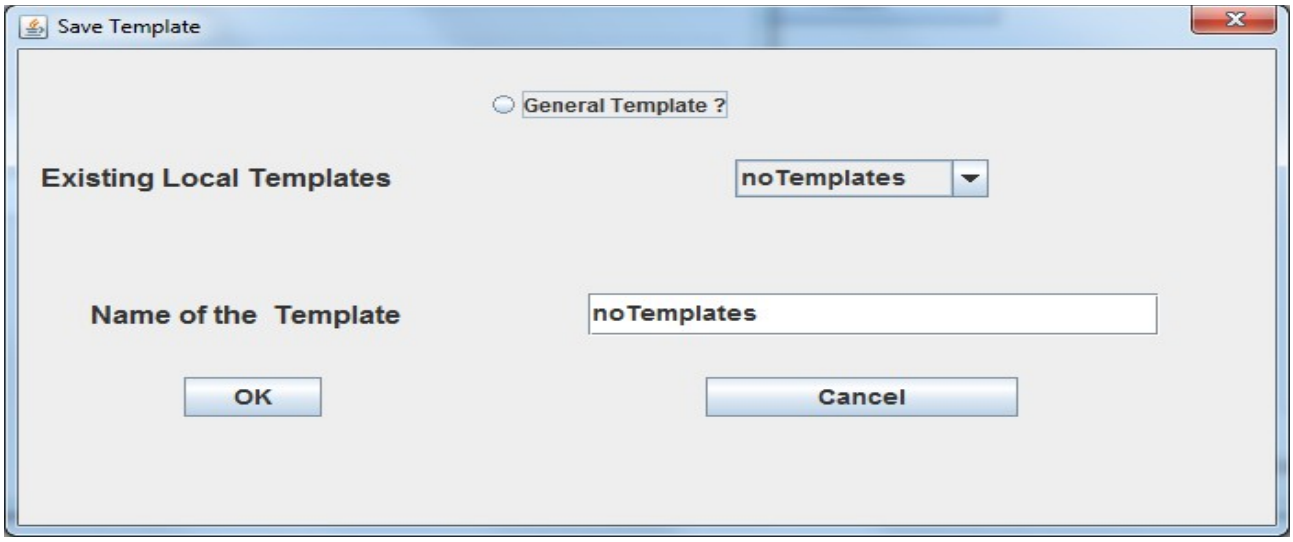
There are two places where templates can be saved :

- general, under the directory <aspectsPerf_Home>/templates/aspectPerf/, available for all project
- local ,under the directory <workspace>/<project>/templates/aspectPerf/, available only for the current project

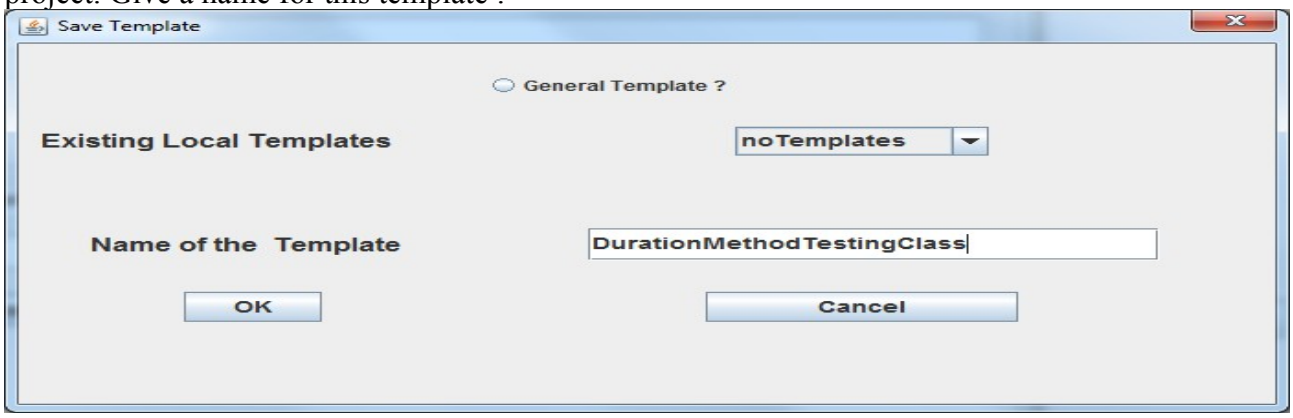
We re-play the screens to configure an Aspect:



click on 

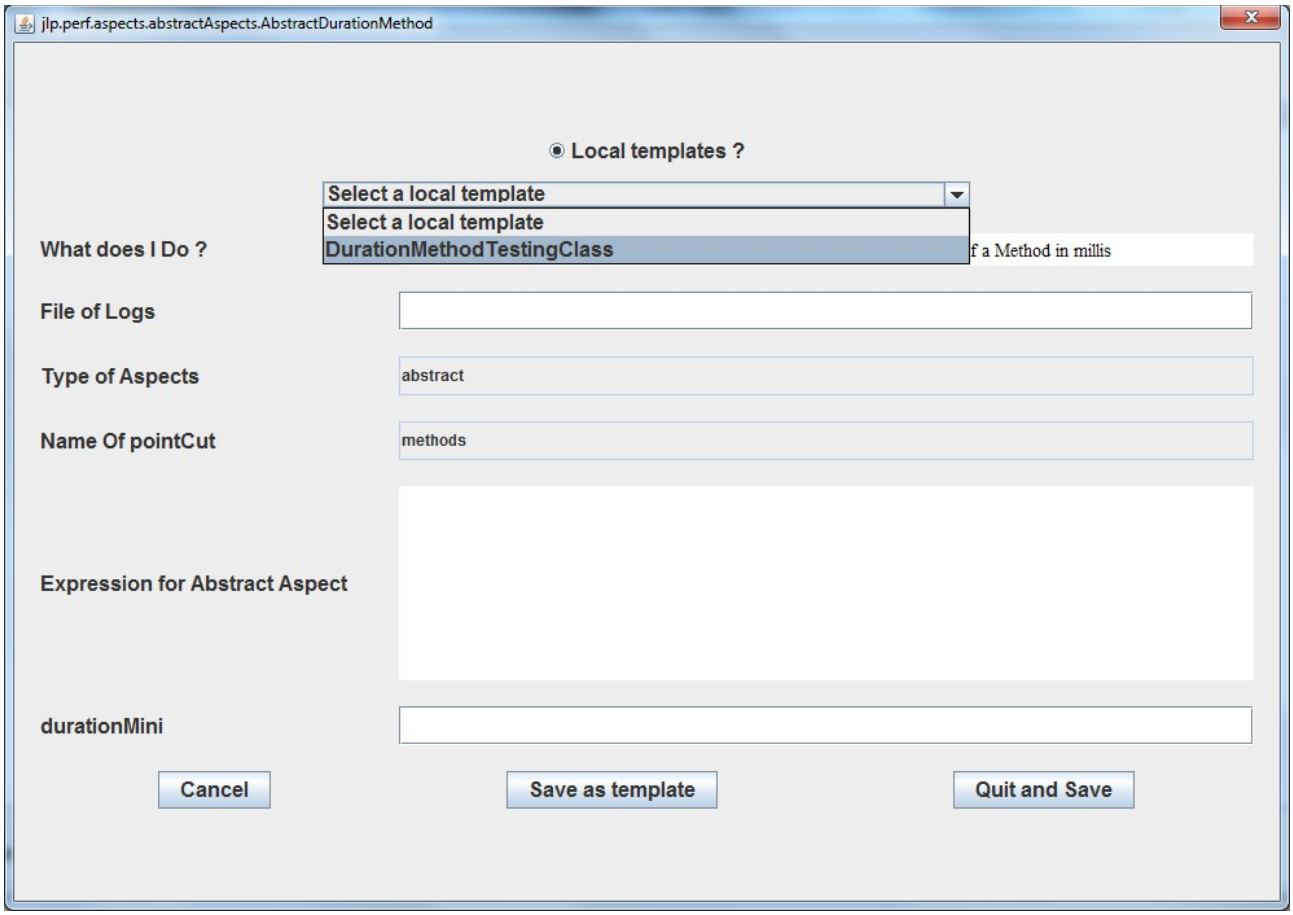


With the radio-button General template ? not selected, the template will be saved for the local project. Give a name for this template :

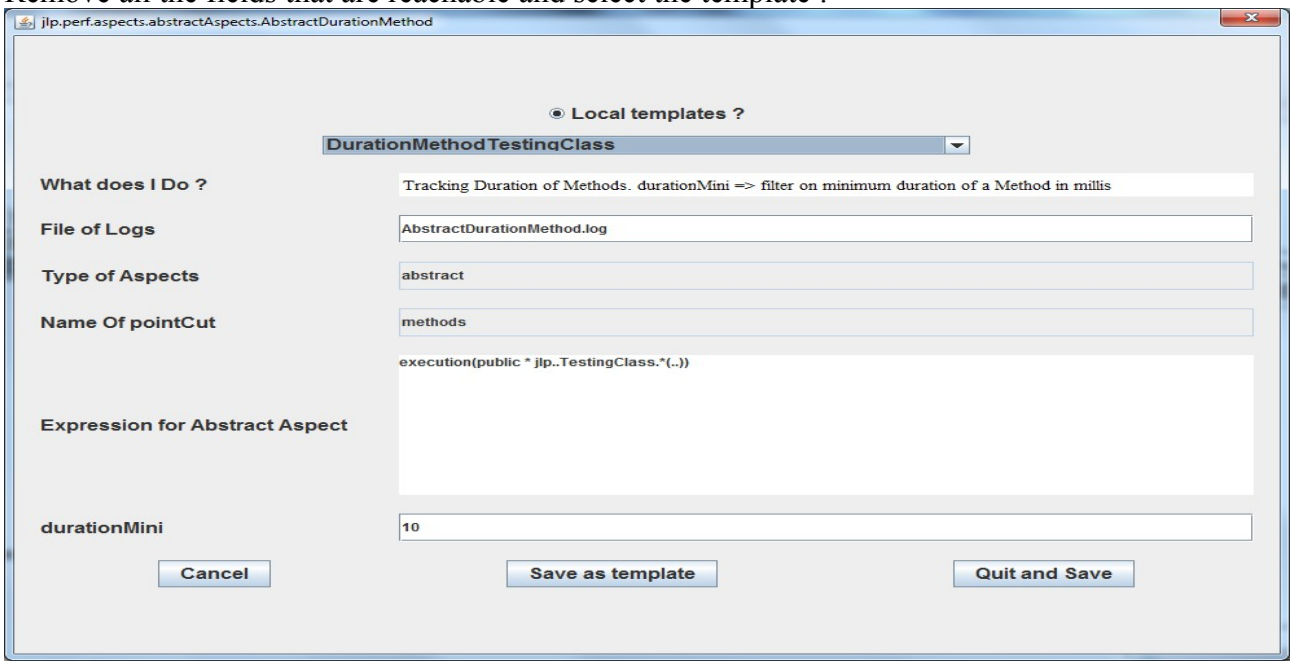


and click OK

When you return on the configuration of the Aspect, you can find it in the list of local template.

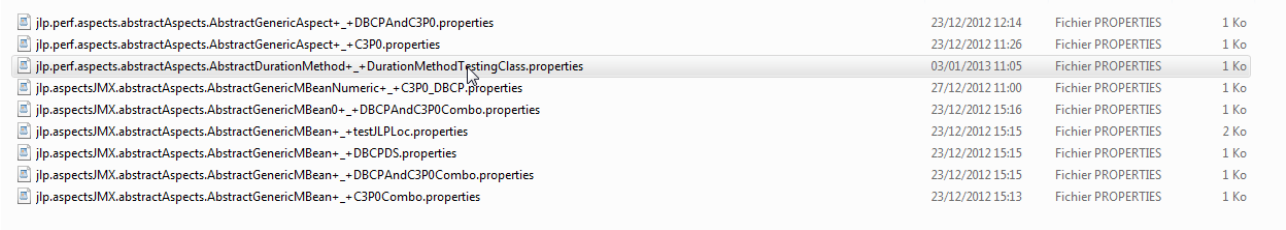


Remove all the fields that are reachable and select the template :



the fields are automatically filled.

Note that the names of the templates are tied with the name of the chosen aspect, the name of the templates are shown below :

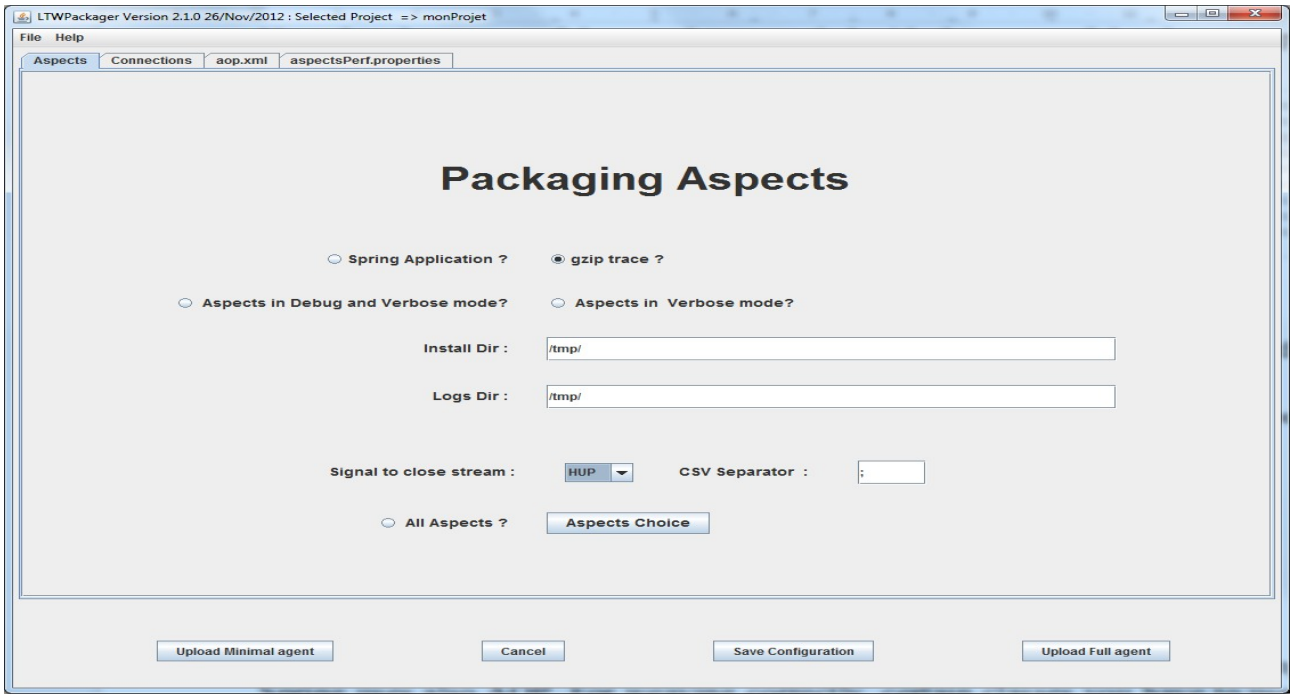


jlp.perf.aspects.abstractAspects.AbstractGenericAspect+_+DBCPAndC3P0.properties	23/12/2012 12:14	Fichier PROPERTIES	1 Ko
jlp.perf.aspects.abstractAspects.AbstractGenericAspect+_+C3P0.properties	23/12/2012 11:26	Fichier PROPERTIES	1 Ko
jlp.perf.aspects.abstractAspects.AbstractDurationMethod+_+DurationMethodTestingClass.properties	03/01/2013 11:05	Fichier PROPERTIES	1 Ko
jlp.aspectsJMX.abstractAspects.AbstractGenericMBeanNumeric+_+C3P0_DBCP.properties	27/12/2012 11:00	Fichier PROPERTIES	1 Ko
jlp.aspectsJMX.abstractAspects.AbstractGenericMBean0+_+DBCPAndC3P0Combo.properties	23/12/2012 15:16	Fichier PROPERTIES	1 Ko
jlp.aspectsJMX.abstractAspects.AbstractGenericMBean+_+testJLPLoc.properties	23/12/2012 15:15	Fichier PROPERTIES	2 Ko
jlp.aspectsJMX.abstractAspects.AbstractGenericMBean+_+DBCPDS.properties	23/12/2012 15:15	Fichier PROPERTIES	1 Ko
jlp.aspectsJMX.abstractAspects.AbstractGenericMBean+_+DBCPAndC3P0Combo.properties	23/12/2012 15:15	Fichier PROPERTIES	1 Ko
jlp.aspectsJMX.abstractAspects.AbstractGenericMBean+_+C3P0Combo.properties	23/12/2012 15:13	Fichier PROPERTIES	1 Ko

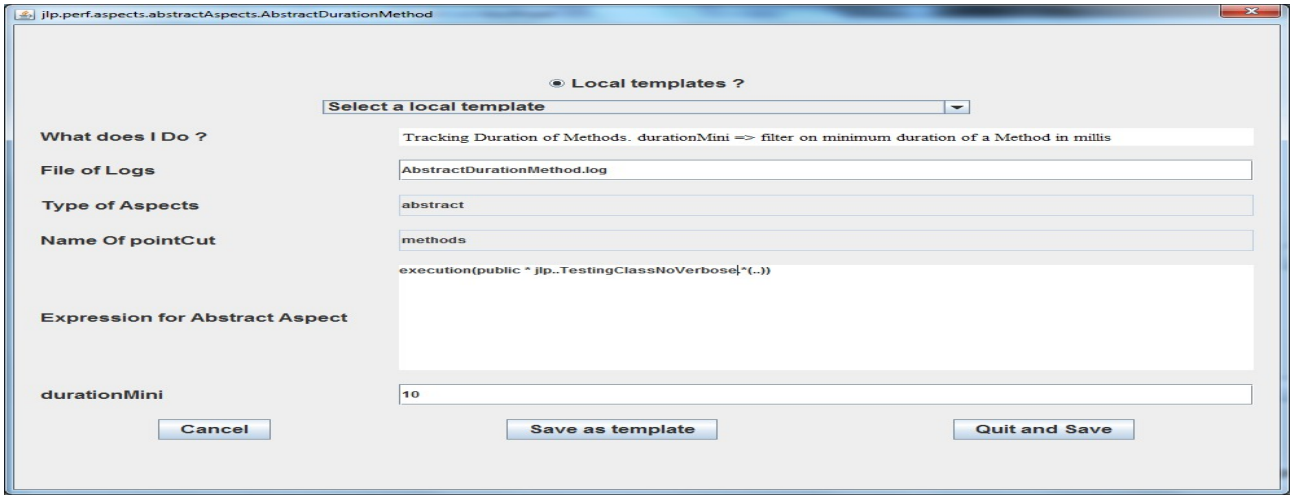
3.5 Generating logs without stopping the JVM

As seen below, we have defined a signal (INT, HUP, USR1...), the goal of this paragraph is to learn to use it.

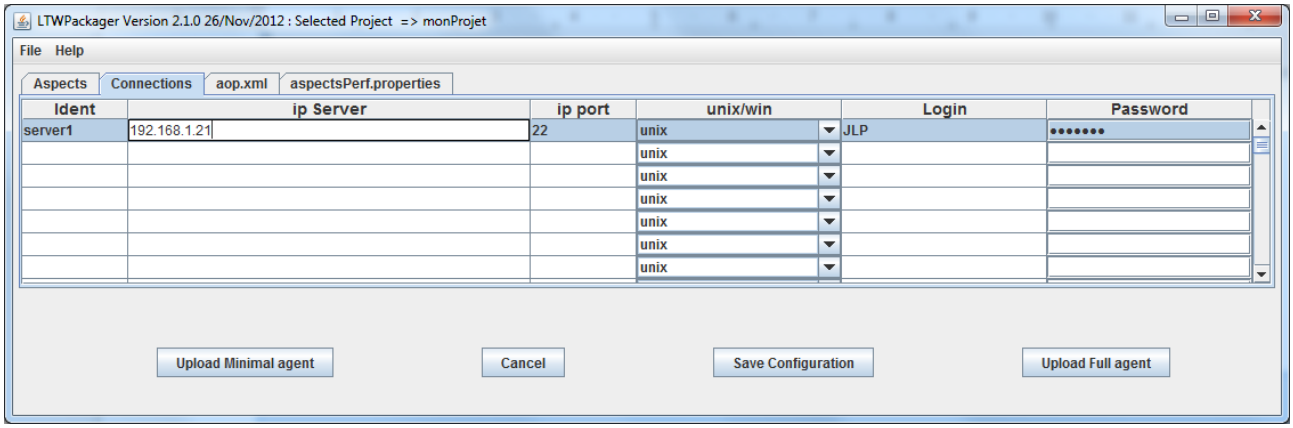
This feature is really useful in linux box and when you choose to log as zipped file =>



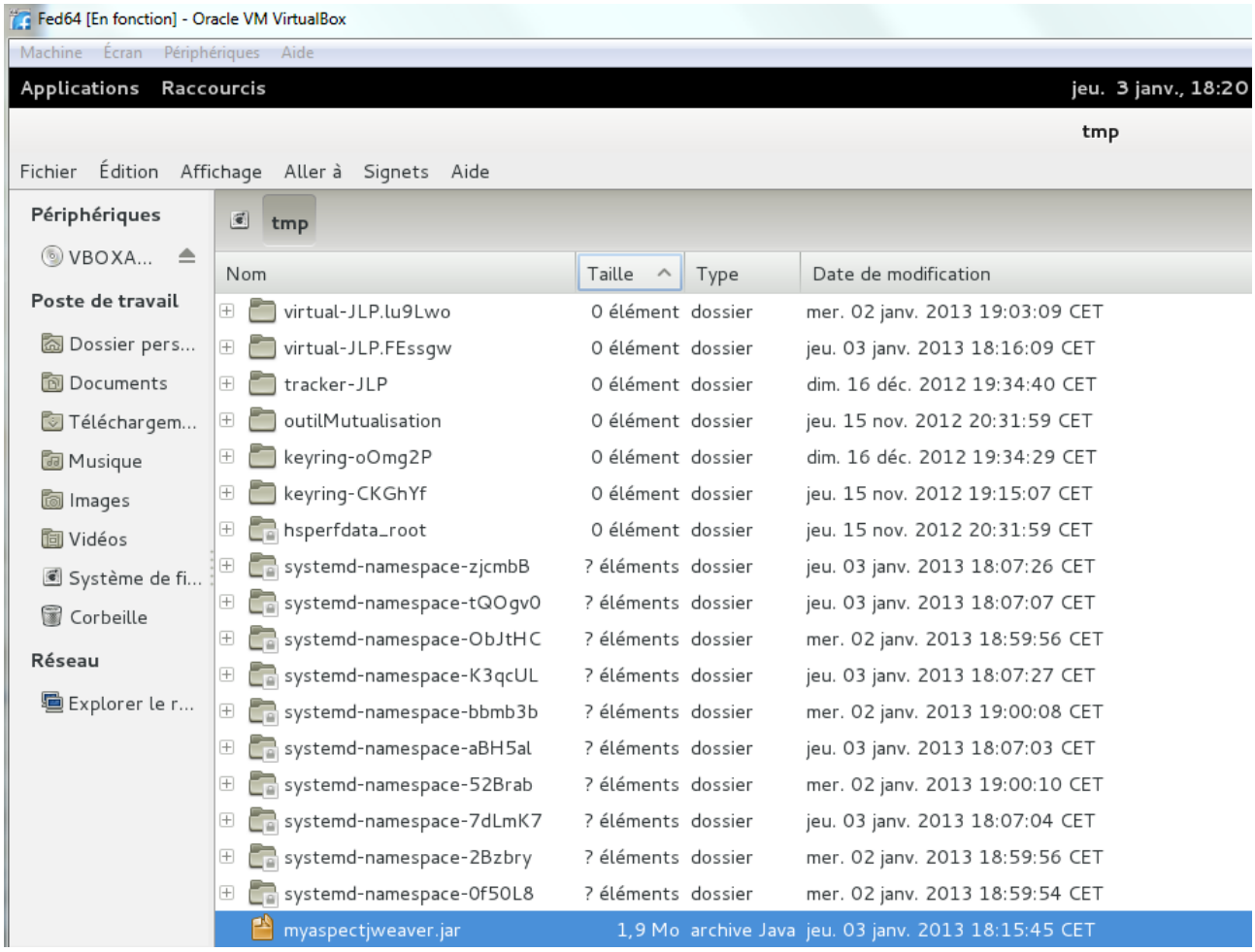
We use the same Aspect with the test class TestingClassNoVerbose (no System.out.println)=>



I have a linux Virtual host on my desktop at address 192.168.1.21



I upload the packaged agent, which contains also the TestingClassNoVerbose



Lauching the application :

```
JLP@localhost:/tmp
Fichier  Édition  Affichage  Rechercher  Terminal  Aide
[JLP@localhost tmp]$ java -javaagent:/tmp/myaspectjweaver.jar jlp.aspectj.test.T
estingClassNoVerbose 20 1000000
Chargement fichier properties
Chargement fichier properties dans le jar
[Instrumentation Aspects by AspectsPerf ] Deb Creation Trace os.name = LINUX
AspectsPerf Version : 2.1.0 of 21/Dec/2012. Author JL Pasturel
[Instrumentation Aspects by AspectsPerf ] jlp.perf.aspects.abstractAspects.Trace
 nomFileLogs =  /tmp/AbstractDurationMethod_20130103_183204.log
[Instrumentation Aspects by AspectsPerf ] jlp.perf.aspects.abstractAspects.Trace
 title =  ####time;Class.methods with  Arguments;time in millisecondes(ms)

sh n est pas null
Before handle signal
After handle signal
[Instrumentation Aspects by AspectsPerf ] Signal Handler initialized by signal :
 |HUP|
```

The aspectj is woven, and file logs is created under /tmp/

```
(Vide)
AbstractDurationMethod_20130103_183204.log.gz   62,4 ko  archive gzip  jeu. 03 janv. 2013 18:35:52 CET
```

As it is a gzip stream, to obtain a readable gzip file, you may close correctly the stream with the configured signal (HUP in our case).

Retrieve the java PID (with ps or jps) =>

```
JLP@l
Fichier  Édition  Affichage  Rechercher  Ter
[JLP@localhost ~]$ jps
2161 TestingClassNoVerbose
2229 Jps
[JLP@localhost ~]$
```

Send the HUP signal to the java process

```
[JLP@localhost ~]$ jps
2161 TestingClassNoVerbose
2229 Jps
[JLP@localhost ~]$ kill -HUP 2161
[JLP@localhost ~]$
```

What happens on the application side ? =>

```
[Instrumentation Aspects par AspectsPerf ] Received signal: SIGHUP
AspectsPerf Version : 2.1.0 of 21/Dec/2012. Author JL Pasturel
[Instrumentation Aspects by AspectsPerf ] Closing all streams and re-opening new
 OutPutStreams
]
```

And under /tmp/ directory

```
AbstractDurationMethod_20130103_183204.log.gz    145,7 ko  archive gzip  jeu. 03 janv. 2013 18:40:47 CET
AbstractDurationMethod_20130103_184047.log.gz    30,9 ko  archive gzip  jeu. 03 janv. 2013 18:42:39 CET
```

The underlined file is correctly close and can be read, a new file is created and it is been filled, the application is still running.

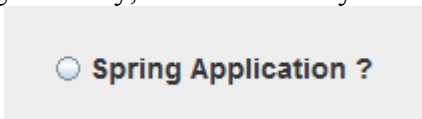
This feature is interesting to skip all useless weaving when starting a WAS or an application, you can focus only when the application or the WAS is steady.

Note that stopping the application CRTL-C (if it is not launched in background with nohup and & parameter)on the shell windows also correctly close the gzip streams (Kill -9 may not close correctly the streams ,it depends on linux box and JDK, because the signal handling is not in the standard package of rt.jar => the classes are located in sun.* packages)

3.6 Known difficulties (Spring, OSGi)

3.6.1 Spring agent

Sometimes, the normal agent aspectjweaver.jar, can't weave poincut in Spring application, because Spring uses also AOP. For weaving correctly, certain classes you have to use the spring agent

(Select the radio button Spring => 

), the name of the generated agent is **springmyaspectjweaver.jar)** . You must also modify the descriptor of the Spring application as shown in the documentation : http://static.springsource.org/spring/docs/3.0.0.RC2/reference/html/ch07s08.html#aop-aj-ltw and following pages.

I have not heavy tested with Spring, so the correct behavior is not guaranteed.

3.6.2 OSGi

Some WAS (JonAS 5+ for example) or standalone applications, have an OSGi architecture. OSGi bundles have each one their own Classloader, that are reachable only from the system Classloader.

So all the classes of the java-agent and the aspect must be loader first by the system classloader.

For example JONAS 5+ based on Apache Felix Engine the two files :

$JONAS_BASE/conf/osgi/defaults.properties

```
bootdelegation-packages com.sun.corba, \
                        com.sun.corba.*, \
                        com.ibm.CORBA, \
                        com.ibm.CORBA.*, \
                        com.sun.org.apache.xalan.internal, \
                        com.sun.org.apache.xalan.internal.*, \
                        com.sun.org.apache.xerces.internal, \
                        com.sun.org.apache.xerces.internal.*, \
                        com.sun.org.apache.xml.internal, \
                        com.sun.org.apache.xml.internal.*, \
                        com.sun.org.apache.xpath.internal, \
                        com.sun.org.apache.xpath.internal.*, \
                        com.sun.jndi.cosnaming, \
                        com.sun.jndi.cosnaming.*, \
                        com.sun.jndi.ldap, \
                        com.sun.jndi.url, \
                        com.sun.jndi.url.*, \
                        com.sun.security.auth, \
                        com.sun.security.auth.*, \
                        com.sun.image, \
                        com.sun.image.*, \
                        org.apache.xalan, \
                        org.apache.xalan.*, \
                        org.apache.xerces, \
                        org.apache.xerces.*, \
                        org.apache.xpath.jaxp, \
                        org.apache.xpath.jaxp.*, \
                       org.apache, \
                       org.apache.*, \
                        jlp, \
                        jlp.*, \
                       org.aspectj, \
                       org.aspectj.*
```

bootdelegation-packages is a standard parameter in OSGi specification (not tested with Equinox Engine)

$JONAS_BASE/conf/osgi/gateway.properties

```
org.osgi.framework.bundle.parent app
```

This parameter is also a standard parameter :

`org.osgi.framework.bundle.parent` - Specifies which class loader is used for boot delegation. Possible values are: `boot` for the boot class loader, `app` for the application class loader, `ext` for the extension class loader, and `framework` for the framework's class loader. The default is `boot`

For others WAS or Applications or Equinox Engine, all this must be certainly adapted.

4 Extending aspectsPerf

4.1 Types of Aspects

There are two types of aspectj used in this tool :

- Aspects that log in a file (gzipped or not)
- Aspects that expose parameters as JMX MBean

Each type can be also abstract (more generic) or concrete when there is no need of generic feature.

We will examine the two type with abstract Simple Aspect example.

4.2 Aspect logging in a file

The example is the Aspect used above :

`jlp.perf.aspects.abstractAspects.AbstractDurationMethod`

For the complete source see the src directory.

First, set static configuration parameters :

```
private static jlp.perf.aspects.abstractAspects.Trace outDurationMethods;
        private static double durationMini = 0;
        private static Properties props;
        private static String dirLogs,sep=";";
        private static DecimalFormat df=new DecimalFormat("#0.000",new  DecimalFormatSymbols(Locale.ENGLISH));
        private static DecimalFormat dfPercent=new DecimalFormat("#0.0",new DecimalFormatSymbols(Locale.ENGLISH));

        static {
                Locale.setDefault(Locale.ENGLISH);
                props = jlp.perf.aspects.abstractAspects.AspectsPerfProperties.aspectsPerfProperties;
                if(props.containsKey("aspectsPerf.default.sep"))
                {
                        sep=props.
                        getProperty("aspectsPerf.default.sep");
                }

                if(props.containsKey("aspectsPerf.default.dirLogs"))
                {
                        dirLogs = props.
                getProperty("aspectsPerf.default.dirLogs");
                        if(!dirLogs.endsWith(File.separator))
                        {
                                dirLogs+=File.separator;
                        }
                }
                else
                {
                        dirLogs = "";
                }
                if (props.containsKey("jlp.perf.aspects.abstractAspects.AbstractDurationMethod.filelogs")) {
                        fileTrace =dirLogs+  props
                                        .getProperty("jlp.perf.aspects.abstractAspects.AbstractDurationMethod.filelogs");
                } else {
                        fileTrace = props.getProperty("aspectsPerf.default.filelogs");
                }

                outDurationMethods = new Trace("####time"+sep+"Class.methods with  Arguments"+sep+"time in
```

```
millisecondes(ms)\n",fileTrace);


                    durationMini = Double.parseDouble(props
                                        .getProperty("jlp.perf.aspects.abstractAspects.AbstractDurationMethod.durationMini"));
          }
```

The aim of logging is the Object **Trace**, that define a file, and a title of the file. For this Aspect a specific parameter is **durationMini** that permits to weave methods when duration is >= **durationMini**

These parameters are set in the file **aspectsPerf.properties** seen above in this document.

After the definition of the abstract pointcut :

```
public abstract pointcut methods();
```

Afterward the code of the advice and the snippet code below show the logging

```
Object around(): methods()       {

                long deb = System.nanoTime();
                Object retour = proceed();
                long fin = System.nanoTime();
                double duree=fin - deb;
                if (duree/1000000 >= this.durationMini) {
…
outDurationMethods.append(new
StringBuilder(outDurationMethods.getSdf().format(Calendar.getInstance().getTime()))
        .append(sep)
        .append(thisJoinPoint.getSignature().getDeclaringTypeName())
        .append(".").append(thisJoinPoint.getSignature().getName())
        .append(strBuff.toString()).append(sep)
        .append(df.format(((double)duree/1000000))).append("\n")
        .toString());
...

}
```

4.3 Declaring the Aspect to the tool

To permit that the aspect appears in the list of available aspects in the tool, we have to configure it in the file **<aspectsPerf_HOME>/config/aspects.properties**.

First add the full name of the aspects to the property **names**, **all in the same line**

```
names=jlp.perf.aspects.abstractAspects.AbstractCounterAllInstances;
jlp.perf.aspects.abstractAspects.AbstractDurationMethod;...
```

if it is a preferred Aspects add id also to the property **prefnames** :

```
prefnames=jlp.perf.aspects.abstractAspects.AbstractCounterAllInstances;
jlp.perf.aspects.abstractAspects.AbstractDurationMethod;...
```

To set the screen of the configuration of the Aspects you have to add a specific paragraph for this Aspect :

```
#AbstractDurationMethod

comment.jlp.perf.aspects.abstractAspects.AbstractDurationMethod=<html>Tracking Duration of Methods.
durationMini =&gt; filter on minimum duration of a Method in millis</html>
```

```
jlp.perf.aspects.abstractAspects.AbstractDurationMethod.type=abstract

jlp.perf.aspects.abstractAspects.AbstractDurationMethod.pointcut=methods

jlp.perf.aspects.abstractAspects.AbstractDurationMethod.filelogs=jlp.perf.aspects.abstractAspects.AbstractDur
ationMethod.filelogs

jlp.perf.aspects.abstractAspects.AbstractDurationMethod.param1=jlp.perf.aspects.abstractAspects.AbstractDurat
ionMethod.durationMini
```

This configuration is used to dynamically create the dialog of the configuration of the Aspect.

You can have more than 1 parameters, naming them param2, param3 …

At the right (the value) is the name of the property that you call from your aspect (this property is set in the file **aspectsPerf.properties** when you save / upload the packaging).

The **comment** property must be written in html style by using XML entities in place of special characters as :

< => <

> => >

& => &

“ => "

' => '

The comment must be written in one line.

4.4 Aspects exposed as MBean

This type of aspect is interesting for Classes that have few instances or are singleton like for example pools (Threads, JDBC, …). In others case, they can saturate the MBean server if every instance Object is a specific MBean.

This Aspect can obviously exists also as the previous type and it can log also in a file (you can also mix the behavior MBean + logging aspect).

The principle is simple, you have to create a normal Java Class which is a Simple Java MBean

```
package jlp.aspectsJMX.mbean;

public interface DurationMethodsCPUMBean {
        public String getName();

        public double getAspectDurationTimeMoy();
// other parameters
…
}
```

Implementation of the interface :

```
package jlp.aspectsJMX.mbean;

public final class DurationMethodsCPU implements DurationMethodsCPUMBean {

        private double aspectDurationCPUSysUserMax = 0;
        private double aspectDurationCPUSysUserMoy = 0;
        private double aspectDurationTimeMax = 0;
```

```
        private double aspectDurationTimeMoy = 0;
        private double aspectDurationCPUUserMoy = 0;
        private double aspectDurationCPUUserMax = 0;
        private double aspectDurationTimeCurrent = 0;
        private double aspectDurationCPUSysUserCurrent = 0;
        private double aspectDurationCPUUserCurrent = 0;
        private int aspectCounterExec = 0;
        private double aspectDurationTimeMini = 0;
…
}
```

Create the aspect, using these Mbeans (see on directory scr the full code source):

```
package jlp.aspectsJMX.abstractAspects;
…
import jlp.aspectsJMX.mbean.DurationMethodsCPU;
public abstract aspect AbstractCPUDurationSimpleMethod {
private boolean supports = false;
        private static HashMap<String, DurationMethodsCPU> hmBean = new HashMap<String,
DurationMethodsCPU>();

        private ObjectName name;

        static ThreadMXBean tMB = null;
        static MBeanServer mbs = null;
        static {
                tMB = ManagementFactory.getThreadMXBean();
                Locale.setDefault(Locale.ENGLISH);
                mbs = ManagementFactory.getPlatformMBeanServer();
                /*
                 * outDurationMethods .append(new StringBuffer(
                 * "####time;Class.methods;time in millisecondes\n") .toString());
                 */

        }
public abstract pointcut methods();

        Object around(): methods()       {
…
// registrer the MBean
name = new ObjectName(strObjName);
mbean = new DurationMethodsCPU();
hmBean.put(strObjName, mbean);
mbs.registerMBean(mbean, name);
…
//fill the attributes
if (mbean.getAspectDurationTimeMax() < duree) {
        mbean.modAspectDurationTimeMax(duree);
}
if (mbean.getAspectDurationTimeMini() > duree) {
        mbean.modAspectDurationTimeMini(duree);
}
if (mbean.getAspectDurationCPUUserMax() < dureeCPUUser) {
        mbean.modAspectDurationCPUUserMax(dureeCPUUser);
}
if (mbean.getAspectDurationCPUSysUserMax() < dureeCPU) {
        mbean.modAspectDurationCPUSysUserMax(dureeCPU);
}
long compteur = mbean.getAspectCounterExec();
                        // traiter les moy
mbean.modAspectDurationTimeMoy((mbean.getAspectDurationTimeMoy() * compteur + duree)/(compteur + 1));
mbean.modAspectDurationCPUSysUserMoy((mbean.getAspectDurationCPUSysUserMoy()
                                        * compteur + dureeCPU)
                                        / (compteur + 1));
mbean.modAspectDurationCPUUserMoy((mbean.getAspectDurationCPUUserMoy()
                                        * compteur + dureeCPUUser)
```

```
                                        / (compteur + 1));

                        // Valeurs courantes
mbean.modAspectDurationCPUSysUserCurrent(dureeCPU);
mbean.modAspectDurationCPUUserCurrent(dureeCPUUser)      ;
mbean.modAspectDurationTimeCurrent(duree);
mbean.modAspectCounterExec(mbean.getAspectCounterExec() + 1);
...
}
```

And after you have to declare this aspect, in the file **aspects.properties** as seen above.