

Online help of the monitoring

The monitoring is a tool to measure and calculate statistics on real operation of an application depending on the usage of the application by the users.

The monitoring is mainly based on statistics of requests and on evolution charts. It allows to improve applications in QA and production and helps to:

- give facts about the average response times and number of executions
- make decisions when trends are bad, before problems become too serious
- optimize based on the more limiting response times
- find the root causes of response times
- verify the real improvement after optimizations



Documentations

[Java SE Reference at a Glance](#)

[Troubleshooting Guide for Java SE 6](#)

[Troubleshooting Guide for Java SE 5](#)

[Monitoring and Managing Java SE 6](#)





[Java SE 6 Performance](#)

[Memory Management](#)



Summary



In the page of monitoring, the summary shows evolution charts for several values of measures.

These measures are taken at a time T, for example every 2 minutes. Each chart follows the evolution of a value of measure on the period short or large chosen with the links above the summary:  the day,  the week,  the month or  the year. In each chart, the mean value in green and the maximum in blue are displayed with numbers below the chart. Charts are persisted: a restart of the application server has no influence on them except of a hole in the measures.

Each chart can be zoomed and resized by clicking on them in the summary.

The charts displayed are:

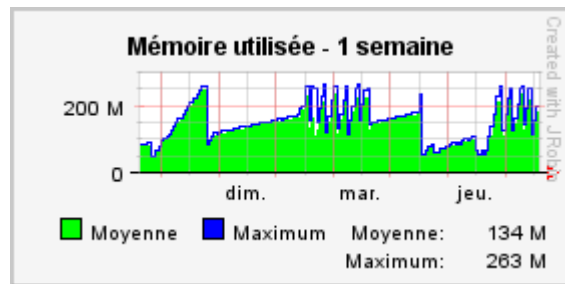
- the java memory used, between 0 and the maximum set by Xmx in the server configuration
- the cpu percentage used by the java process, between 0 and 100 even if there are several cores or several processors
- the number of http sessions (or number of connected users)
- if linux: the system load, the cpu percentage of the garbage collector and the number of opened files
- the number of active threads (or number of current http requests)
- the number of active jdbc connections (or number of current sql requests)
- the number of used jdbc connections (or number of opened sql transactions); in the case where there is no datasource, it is in fact the number of opened jdbc connections in the connections pool
- for each counter of requests (http, sql and possibly ejb or spring) :
 - the number of hits per minute (or number of executions of requests per minute)
 - the mean time in milliseconds
 - the percentage of system errors (these values for a counter are a mean for the last period of measure)

The link ' Update' refreshes the page and the charts. The link ' PDF' displays the report in PDF format for Adobe Reader.

If a collect server is used to display the monitoring of several servers in a farm or in a cluster, the chart of the memory is the sum of the memories in each server, but the cpu percentage is between 0 and 100 for all

servers, and the number of http sessions is the sum of sessions in each servers, as are the number of active threads, the number of active or used jdbc connections, and the number of hits per minute or the mean time in milliseconds.

Memory chart: use case of an application used intermittently (day/night for example)



The memory in the chart increases rapidly when the application is used, and it increases slowly but still increases when the application is not used. A chart with slopes, as in the above example, is classic even if the application is not used.

In all cases and as needed, the memory is finally released at once by the garbage collector (major GC) and goes to a low level, before rising again rapidly or slowly. If the resolution set is sufficiently fine, the chart also shows the minor GCs as small sawtooth between the major GCs. These minor GCs release also some memory but in less quantity than the major GCs. It is possible to force a major GC with the action 'Execute the garbage collector' in the part 'System informations' of the report.

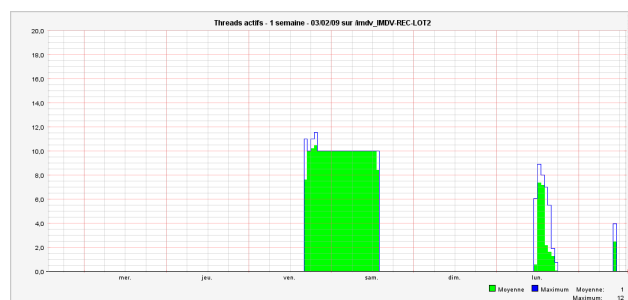
Memory chart: use case of a memory saturation

If the memory in the chart increases to the maximum and if the application can not release some of it with the garbage collector, the server possibly causes some OutOfMemoryErrors to interrupt the execution and to release some memory if possible.

Until the memory is sufficient, the cpu remains at 100% because the garbage collector runs permanently. This can cause extreme slowness or a blocking of the application server.

The solutions can be to increase the maximum java memory (parameter Xmx in configuration of server), and possibly the physical memory of the server, or to optimize if possible the memory used by the application.

Charts of active threads and of active jdbc connections: use case of a flat level (long requests)



When a application is sparsely or not used, the charts of active threads and of active jdbc connections stay at 0. Indeed, most requests are fortunately short and most measures (done each 2 minutes for example) are taken without active requests, except 1 or 2 occasionally shown as a peak.





But, when there is a flat level in the chart of active threads like in the example above, this means that there is one or several requests which are long (several minutes or several hours) and which potentially saturate the application server or the database.

To find the cause:

- If the flat level of active threads is reproduced exactly in active jdbc connections, this shows that the cause is in one or several sql request(s) and not in the execution of java code.
- If the flat level(s) continues currently, you can find the http (and sql) requests involved in the '🕒 Current requests' part of the report.
- If the flat level(s) has ended, you can find the statistics of http (and sql) requests involved, in the detailed tables of '🌐 Http (and sql) statistics', except if the application server has been stopped before the end of these requests.
- The drill-down display in the detail of an http request can help you to find the root cause in the façades or the sql requests.
- It can be useful to check with the monitoring the state of the memory and the cpu used by the application server. It can also be useful to check on your own the state of the memory, of the cpu and of disk access of your database.

Statistics

The statistics of a counter shows the statistics of the requests executed on the application server. 4 counters of requests exist today:

-  Http statistics
-  Sql statistics (jdbc)
-  Ejb statistics (if ejb3 façades on JavaEE 5)
-  Spring statistics (if spring façades)

For each counter, a request appears in the statistics when it is finished; to know the unfinished requests see the '🕒 Current requests' part of the report.

As for the charts, the statistics of requests are for the period small or large chosen with the links above the charts: the day (since 0h), the week, the month or the year. So it is possible to see the statistics for only the current day, or for example for a version of the application deployed since a month without the statistics of previous versions.

And for each counter, the report shows a summary of statistics for all the requests (global), of the statistics going beyond the threshold of alert level 1 (warning) and of those going beyond the threshold of alert level 2 (severe). The details of the statistics for each request are displayed by clicking the link '+ Details'.

Each statistic of request shows:

- the content of the request
- an evolution chart for mean times of this request in tooltip for the same period as the statistics; this chart can be zoomed by clicking on the content of the request
- the percentage of cumulated time (mean time * nb of hits) compared to other requests
- the number of hits (or number of executions)
- the mean time in milliseconds
- the maximum time in milliseconds
- the standard deviation between execution times: if it is high, the execution times are varied, and if it is low the execution times are closed to one another
- the percentage of system errors

and if it is an http statistic:

- the mean size in kilo-bytes of the response stream
- the mean number of sql hits, by execution of http request
- the mean time executing sql requests, by execution of http request

Like all tables of the report, the tables of statistics can be ordered in ascending or descending order by clicking on headers of columns.

The statistics are persisted for each counter: a restart of the application server has no influence on them. If a collect server is used to display the monitoring of several servers in a farm or in a cluster, the statistics of counters are global for all servers.

Note: In MS Windows (except Vista), the clock resolution used is about 16 ms (either 0, either 16, or 32 ms). So the time of an execution is not accurate below 16 ms with Windows. This is in part corrected by use of a mean when there are sufficient executions. But this is not grave because in general, the problematic requests will have an execution time much longer than 16 ms.

Statistics of http system errors

The statistics of http system errors shows the errors which come back to the http filter of the monitoring, either by exceptions thrown by the application via a servlet, either by an http error code (404 "not found" or 500 "internal server error" for example, [complete list](#)). These statistics display the list of the 250 most frequent system errors with the number of cases of each error on the chosen period, and with the mean time of the request for each error. Only the most frequent error is displayed initially, the other errors being displayed by clicking the link '+ Details'. The dates, hours, users and complete http requests of the last 100 errors are displayed by clicking the link '+ Last errors'. By clicking on the name of an error, it is possible to see for this error the evolution chart of the number of cases over time and the java stack-trace of the error when it is a java exception.

These statistics of errors allow to improve the reliability of the application based on its real usage in production.

Statistics of system errors logs

The statistics of system errors logs show the 'warning' and 'error' logs written by the application (the libraries log4j of apache and java.util.logging of the jdk are both supported). These statistics display the list of the 500 most frequent errors logs with the number of cases of each error on the chosen period. Only the most frequent error log is displayed initially, the other logs being displayed by clicking the link '+ Details'. The dates, hours, users and where appropriate the complete http requests of the last 100 errors logs are displayed by clicking the link '+ Last errors'. By clicking on the name of an error log, it is possible to see for this log the evolution chart of the number of cases over time and the java stack-trace of the error when a java exception has been written in the log.

These statistics of errors logs allow equally to improve the reliability of the application based on its real usage in production.

Current requests

The current requests shows at the moment of the generation of the report the executions of not ended requests.

The tree of http requests, possibly ejb, spring and/or sql is displayed with for each of the requests the time elapsed already, the cpu time elapsed, the number of sql requests already executed and the sql time of these requests.

The global statistics of the requests are displayed again for the current requests: mean time, mean cpu time, mean sql hits and mean sql time. It allows to compare the current requests with the global statistics. The current java stack-trace is displayed by a tooltip on the thread (if jdk 1.6).

Only the longest request is displayed initially, the others are displayed by clicking on the link '+ Details'.

System informations








The system informations display at the moment of the generation of the report some informations on the java server, its state and on the operating system of the server.

The main informations displayed initially are: used java memory, number of http sessions, number of

active threads, number of active jdbc connections and number of used jdbc connections. These are the same values which are used for the charts.

The other informations, such as the version of the server, of the operating system or of the database or the memory of the operating system, are displayed by clicking on the links '+ Details'.

In this part of the system informations, some links allow to run the system actions:

-  'Execute the garbage collector', to force a release of memory
-  If jdk 1.6, 'generate a heap dump', to dump all the content of the memory in a file of the temporary directory on the server, which you can open with [VisualVM](#) of the JDK or with [Eclipse MAT](#)
-  'Invalidate http sessions', to force all users to disconnect
-  'View http sessions', to see the attributes, the serialized sizes (generally higher than the sizes in memory), the country and possibly the user of each session (if authentication by JavaEE)
-  If jdk 1.6, 'view memory histogram', to display the number of instances in memory for each java class
-  'View deployment descriptor' : web.xml file of the application
-  'View OS processes' : list of processes of the operating system (linux with [ps](#) or windows with [tasklist](#)) and for each process: user, memory and cpu

If a collect server is used to display the monitoring of several servers in a farm or in a cluster, the system informations of each server are displayed (aggregated for the list of sessions and for the memory histogram) and the actions are executed on each server one at a time.

Threads

This table displays the list of all threads in the server, with for each the priority, the state and the stack-trace in a tooltip (if jdk 1.6) among other.

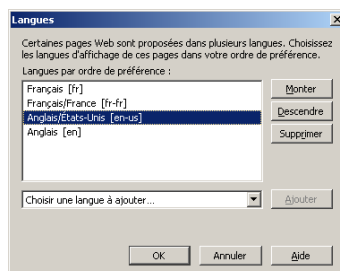
Caches of data

The caches of data display the list of caches in the java application provided that the library [ehcache](#) is used for them.

For each of the caches, some statistics are displayed: the number of objects in memory and on disk at the moment, the efficiency of the memory cache (percentage since the start of the server of the memory cache hits compared to the disk cache hits), the global cache efficiency (percentage of the memory cache or disk cache hits compared to all cache access including misses for data not in cache). The configuration of each cache is also displayed for information.

Language

The monitoring is displayed in French or in English depending of the preferred language in your browser. If the monitoring is in French language instead of being in English language, you should modify the preferred language. For example, in [Firefox](#), open the menu and popup "Tools, Options, Content, Language ... Choose" and move "Anglais [en]" to the first place:



Icons credits

[Silk, mini and flag icons \(Creative Commons\)](#)

[Tango icons \(GPL\)](#)