

Baptiste Wicht



617 followers

## Related posts

1. Discover Java VisualVM 1.3(discover-java-visualvm-1-3.html)
2. Tip : Profile an OSGi application with VisualVM(tip-profile-osgi-application-visualvm.html)
3. How to profile C++ application with Callgrind / KCacheGrind(../2011/09/profile-c-application-with-callgrind-kcachegrind.html)
4. How to profile your applications using the Linux perf tools(../2011/07/profile-applications-linux-perf-tools.html)
5. Develop a modular application with JTheque Core 2.0.3(../02/modular-application-jtheque-core-2-0-3.html)

## Recent comments



(<http://disqus.com/by/sganesh99/>)  
 gansai(<http://disqus.com/by/sganesh99/>)  
 you might not have imported Semaphore in your...

Java Concurrency – Part 4 : Semaphores(<http://baptiste-wicht.com/posts/2010/08/java-concurrency-part-4-semaphores.html>) · 19 hours ago(<http://baptiste-wicht.com/posts/2010/08/java-concurrency-part-4-semaphores.html#comment->

# Profile your applications with Java VisualVM

Baptiste Wicht — 2010-07-13 01:08 — 12 Comments([profile-applications-java-visualvm.html#disqus\\_thread](http://profile-applications-java-visualvm.html#disqus_thread)) — Source([profile-applications-java-visualvm.wp](http://profile-applications-java-visualvm.wp))

When you need to discover what part of an application consume the more CPU or Memory, you must use a profiler to do that.

One profiler, packed by default with the Sun JDK is Java VisualVM. This profiler is really simple to use and really powerful.



(../wp-content/uploads/2010/07/VisualVM-Startup.png)

In this post, we'll see how to install it and use it to profile an application.

Normally, to install it, you have nothing to do, because, it's installed with the JDK. But in several Unix system, like Ubuntu, this is not the case. If you want to install it, just use apt-get (or aptitude) :

```
sudo apt-get install visualvm
```

To launch it just launch jvisualvm  
 (/visualvm.exe in the bin directory of the jdk for

semaphores.html#comment-1798511438)



Andrew

yeah but what if we would like to ask the...

Solve Einstein's Riddle using Prolog(<http://baptiste-wicht.com/posts/2010/09/solve-einsteins-riddle-using-prolog.html>) · 2 days ago(<http://baptiste-wicht.com/posts/2010/09/solve-einsteins-riddle-using-prolog.html#comment-1794155658>)



(<http://disqus.com/by/wichtounet/>)

Baptiste

Wicht(<http://disqus.com/by/wichtounet/>)

Hi. I'm sorry, but I don't understand what you...

Java Concurrency – Part 7 : Executors and thread pools(<http://baptiste-wicht.com/posts/2010/09/java-concurrency-part-7-executors-and-thread-pools.html>) · 4 days ago(<http://baptiste-wicht.com/posts/2010/09/java-concurrency-part-7-executors-and-thread-pools.html#comment-1791705023>)



(<http://disqus.com/by/wichtounet/>)

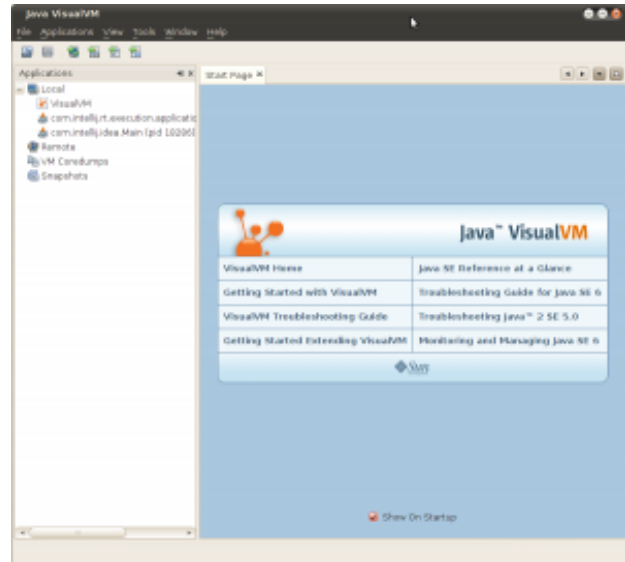
Baptiste

Wicht(<http://disqus.com/by/wichtounet/>)

Hello. I'm sorry, but I don't get what you want...

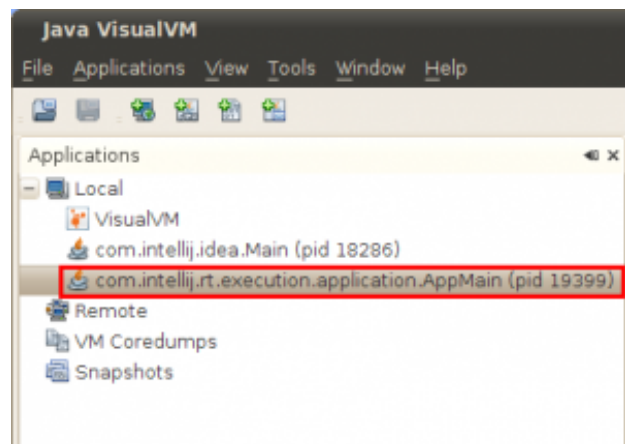
(visualvm.exe in the bin directory of the jdk for Windows).

That will open the following window :



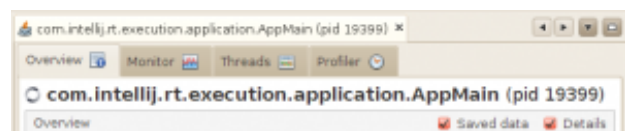
([../wp-content/uploads/2010/07/VisualVM-Start.png](http://wp-content/uploads/2010/07/VisualVM-Start.png))

There is not a lot of interesting things to see here. To profile a application, you just have to launch it and VisualVM will detect it as launched :



([../wp-content/uploads/2010/07/VisualVM-Application.png](http://wp-content/uploads/2010/07/VisualVM-Application.png))

After that, you just have to double click to view information about your running application. You've four tabs available for your applications (Overview, Monitor, Threads, Profiler). We'll see all that 4 tabs. First of all, the default tab, the overview :



Solve Einstein's Riddle using Prolog(<http://baptiste-wicht.com/posts/2010/09/solve-einsteins-riddle-using-prolog.html>) · 4 days ago(<http://baptiste-wicht.com/posts/2010/09/solve-einsteins-riddle-using-prolog.html#comment-1791704363>)



(<http://disqus.com/by/wichtounet/>)  
Baptiste Wicht(<http://disqus.com/by/wichtounet/>)  
Hello, Normally, each of your functions should...

How to profile C++ application with Callgrind / KCacheGrind(<http://baptiste-wicht.com/posts/2011/09/profile-c-application-with-callgrind-kcachegrind.html>) · 4 days ago(<http://baptiste-wicht.com/posts/2011/09/profile-c-application-with-callgrind-kcachegrind.html#comment-1791703777>)

## Blogroll

- AsJava.com : Java Tutorial(<http://www.asjava.com/>)
- Mkyong : Java Tutorials(<http://www.mkyong.com/>)



([../wp-content/uploads/2010/07/VisualVM-Overview.png](http://wp-content/uploads/2010/07/VisualVM-Overview.png))

This tab contains the main informations about the launched application. You can see the main class, the arguments of the command line, the JVM arguments. You can also see what kind of JVM is running your program and where the JVM is located. And you can see all the properties set in the program.

A more interesting tab is the "Monitor" tab :

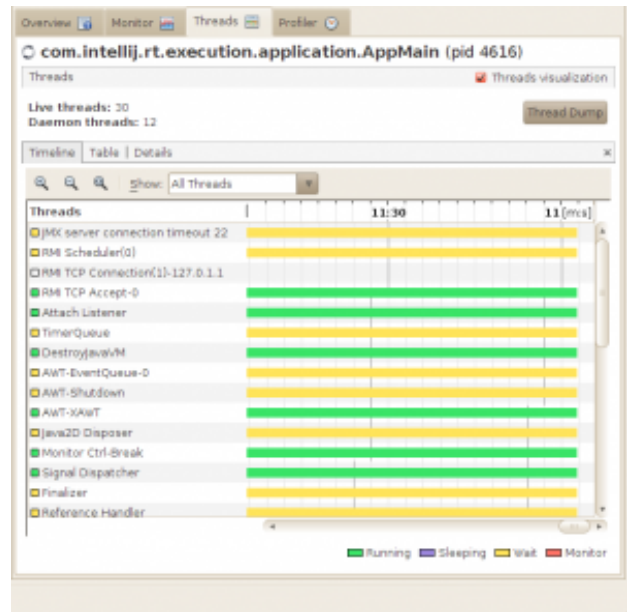


([../wp-content/uploads/2010/07/VisualVM-Monitor.png](http://wp-content/uploads/2010/07/VisualVM-Monitor.png))

This tab follow the CPU and Memory usages of your applications. You have 4 graphs in this view. The first one, from left to right, up to down, display the CPU usage and the Garbage Collector CPU usage. The second graph display the usage of the heap space and the PermGen space. The next graph display the total number of classes loaded in the application and the last one displays the number of threads currently running. With these graphs, you can see if your application take too CPU or if there is too memory used by

take too often or if there is too memory used by the application.

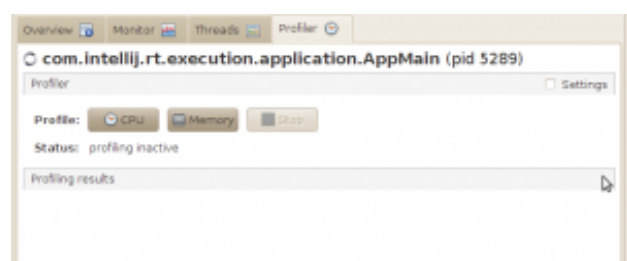
The third tab provide some details about Threads :

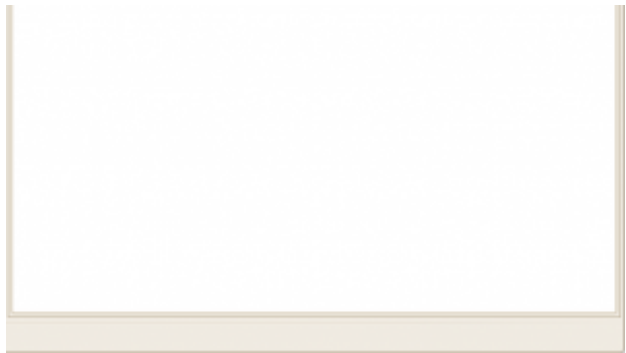


(../../../../wp-content/uploads/2010/07/VisualVM-Threads.png)

In this view, you can see how the different threads of the application are changing state and how they evolve. You can also see the time each time pass in each state and you can have details about the threads you want.

And now, I think the most interesting tab, is the Profiler one :





(../../../../wp-content/uploads/2010/07/VisualVM-Profiler.png)

When you open this tab first, it contains no information at all. You must start one kind of profiling before seeing informations. We'll start with CPU profiling. Just click on the CPU button the instrumentation will start. During the instrumentation, the application will be blocked. After the instrumentation, you can access the application again and you will see the results of the profiling be displayed in the table. Of course the profiling has an overhead on your application. Normally it's not visible, but for certain applications, you can loose a lot of fluidity. Here are the results I have obtained with my simple application :

com.intelij.rt.execution.application.AppMain (pid 5065)

Profiler

Profile: CPU Memory Stop

Status: profiling inactive

Profiling results

Hot Spots - Method	Self time [%]	Self time	Invocations
poletse.grp6.Zeus.waitForTimeout ()	81.6%	329891 ms	6509
javax.swing.Timer\$DoPostEvent.run ()	5.3%	21549 ms	1414
poletse.grp6.BrainConnector.notifyDecision	4.7%	18920 ms	6509
poletse.grp6.Zeus.getSensor (poletse.gr...	3.7%	6803 ms	6509
javax.swing.SystemEventQueueUtilities\$Com...	1.3%	5185 ms	1429
sun.awt.image.ImageFetcher.run ()	1.2%	5003 ms	1
poletse.grp6.Point.minDistance (java.awt...	1.1%	4376 ms	3155113
poletse.grp6.Point.minRaw (double, doubl...	0.9%	3781 ms	6396689
poletse.grp6.Point\$RawDirection.<init> (d...	0.9%	3439 ms	6396689
poletse.grp6.Torus.getInstance ()	0.4%	1775 ms	6489839
poletse.grp6.Point\$RawDirection.<init> (d...	0.4%	1704 ms	6396689
poletse.grp6.Poietse.getPotential ()	0.1%	234 ms	306987
poletse.UISimulationPanel\$2.run ()	0%	192 ms	1346
poletse.grp6.Poietse.getCenter ()	0%	105 ms	224167
poletse.grp6.Ellipse.contains (poletse.gr...	0%	103 ms	201752
poletse.grp6.BrainForm\$BrainFormAction.f...	0%	69 ms	6509

[Method Name Filter]

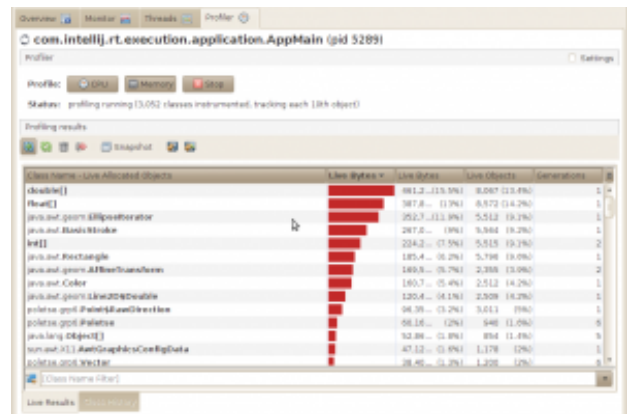
(../../../../wp-content/uploads/2010/07/VisualVM-Profiler-CPU.png)

In my example, we can see that the waitForTimeout method takes 81.6% of the CPU time. We can also see that the notifyDecision and getSensor methods are the two next most CPU consuming methods,

perhaps, it will be interesting to optimize them.

You can also look at the number of invocations of each, perhaps, you'll find a method that is invoked too much time.

The next profiling we can do is the Memory profiling. Here again, you have to start the profiling and the instrumentation will start and during this, the application will be frozen. Here are the results for my application :



(../../../../wp-content/uploads/2010/07/VisualVM-Memory.png)

Here we can see that this application store some big double[] and float[] arrays and that EllipseIterator and BasicStroke classes take also a lot of memory spaces.

In both the memory and CPU profiling, you can save the results to a file to see it later. By example, you can let an application working the all night, save the results the morning and examine them or make three profiling and compare the three.

To conclude, I have to say that this profiler is really simple but also really powerful to use. We've the main features we want for a profiler and the results are really good. That kind of tools can really help you to improve an application to use less CPU and memory. Of course that kind of tool doesn't do everything, it just help showing what part of the application must be improved, the improvement part is the task of the developer and it's not the easiest one. But having that kind of tool is a good start.

**Java**(../../../../categories/java.html)

**Performances**(../../../../categories/performances.html)

**Tools**(../../../../categories/tools.html)

Previous post(version-control-with-git-book-review.html)

Next post(state-of-the-lambda.html)

# Comments

## AROUND THE WEB

## WHAT'S THIS?

Naturalon

Top 10

Cancer

Causing

Foods that

are NOW

Sitting in

Your Fridge

MAX Workouts

The 3 Worst

Things that

Age You

Faster

(Avoid)

Moneynews

Social

Security: How

To Get

\$1,000 More

a Month

Lifefactopia

How New

iPads are

Selling for

Under \$40

## ALSO ON @BLOG("BAPTISTE WICHT")

New WordPress  
Plugin – Google

Linux tip: Force  
systemd networkd

Why and how I  
completely left

Home Server  
Adventure – Step 1

Comments

Community

Login

Sort by Best

Share Favorite

Join the discussion...



Patilamol1687 • 3 years ago

Thank , nice application





1 ^ | v • Reply • Share ›

**Rug Cleaning Plantation**

• 4 years ago

nice application .thanks for the post .

1 ^ | v • Reply • Share ›

**Mold Removal Sunrise**

• 4 years ago

Great information. nice way of telling this process.

1 ^ | v • Reply • Share ›

**Anand Kumar** ➔ Mold

Removal Sunrise

• 4 years ago

Its good way, but still it requires more information with partical or viedo to show how to process memory thread dumps

^ | v • Reply • Share ›

**Javin Paul** • 3 years ago

Thanks for hard work. Visual VM is Indeed an useful tool to find memory leaks and fix [OutOfMemroyError](#) in java only thing its not integrated with IDE like Netbeans profiler ,let me know if there is a way to integrate with IDE like Eclipse and Netbeans than it would be just great.

^ | v • Reply • Share ›

**Alan Jobs** ➔ Javin Paul

• a year ago

You can integrate it with Eclipse IDE using the link:  
<http://visualvm.java.net/eclip..>

^ | v • Reply • Share ›

**Mold Removal Illinois**

• 4 years ago


Very good description.

^ | v • Reply • Share ›

**Vickyb20** • 4 years ago

Can i profile a web application usina





visualVm


^

|

▼

• Reply

• Share ›



Mold removal contractor

• 4 years ago

Th...

Contents © 2014 Baptiste Wicht(mailto:baptistewicht@gmail.com) - Powered by Nikola(http://getnikola.com)



(http://creativecommons.org/licenses/by/4.0/) Source(profile-applications-java-visualvm.wp)