

[Inicio](#) | [Quiénes somos](#) | [Formación](#) | [Publicaciones](#)

Tutoriales.

[adictosaltrabajo](#) / [Tutoriales](#) / [Optimización Java con Eclipse Profiler Plugin](#)

Alejandro Pérez García

Alejandro es socio fundador de Autentia y nuestro experto en Java EE, Linux y optimización de aplicaciones empresariales.

Ingeniero en Informática y Certified ScrumMaster.

[Seguir @alejandropgarcia](#)

Si te gusta lo que ves, puedes contratarle para darte ayuda con soporte experto, impartir cursos presenciales en tu empresa o para que realicemos tus proyectos como factoría (Madrid).

Puedes encontrarme en [Autentia](#): Ofrecemos servicios de soporte a desarrollo, factoría y formación.



Optimización Java con Eclipse Profiler Plugin

julio 21, 2004

Alejandro Pérez García

Sin comentarios

Tutoriales

34969 visitas

Como analizar el rendimiento de nuestras aplicaciones con Eclipse Profiler Plugin

1. Introducción

En este tutorial vamos a ver una poderosa herramienta para analizar el rendimiento de nuestras aplicaciones Java.

Esta herramienta es "Eclipse Profiler Plugin", un plugin de Eclipse que nos va a permitir estudiar a fondo como se distribuye el tiempo de ejecución entre los distintos métodos, clases, ...

A la hora de mejorar el rendimiento de nuestras aplicaciones, es fundamental el uso de una herramienta de este tipo, ya que normalmente el 80% de la pérdida de rendimiento está concentrada en un 10% de código. Con este tipo de herramientas vamos a poder identificar cual es ese 10% de código, y no malgastar recursos optimizando código que puede que sólo se ejecute una única vez.

También hay que tener en cuenta que es mucho mejor no centrarnos en la optimización cuando estamos construyendo la aplicación. Es preferible centrarnos en un buen diseño, y en hacer que funcione, y en una etapa posterior y con la ayuda de este tipo de herramientas localizar los puntos críticos (no tiene sentido estar una

Los más visitados de la semana

[Introducción a NestJS](#)[VirtualBox. Configuración de la conexión de red.](#)[Page Analytics](#)[Instalación y uso del plugin de comentarios de Facebook en n...](#)[Parte 1. Aprendiendo HTML para crear una página web](#)

semana reinventando el algoritmo de ordenación quicksort, para luego descubrir que en la lista vamos a tener un máximo de 5 elementos).

Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información

ACEPTAR

http://eclipsecolorer.sourceforge.net/index_profiler.html

En esta página podemos encontrar bastante información de como instalarlo, y de las características que nos ofrece.

En primer lugar descargamos el plugin de la página: <http://sourceforge.net/projects/eclipsecolorer>. En esta página pincharemos sobre el enlace "Download" de la versión 0.5.31 (la última versión al escribir este tutorial, ojo, es para Eclipse 3.0).

Una vez descargado el archivo `ru.nlmk.eclipse.plugins.profiler_0.5.31.zip` tendremos que descomprimirlo en el directorio de plugins de eclipse. En mi Linux sería:

```
cd /opt/eclipse/plugin
unzip /download/ru.nlmk.eclipse.plugins.profiler_0.5.31.zip
```

Donde `/opt/eclipse` es el directorio donde tengo instalado Eclipse 3.0, y `/download` es el directorio donde me he descargado el archivo `ru.nlmk.eclipse.plugins.profiler_0.5.31.zip`.

Para terminar con la instalación debemos copiar una librería nativa del sistema donde estemos haciendo la instalación (Linux, Windows, ...). Esta librería viene con plugin, así que basta con hacer (en el caso de mi Linux):

```
cd /opt/eclipse3.0/plugins/ru.nlmk.eclipse.plugins.profiler/native
tar -xzf profiler_linux.tgz
cp libProfilerDLL.so /opt/j2sdk1.4.2_04/jre/lib/i386
```

Donde `/opt/j2sdk1.4.2_04` es donde tengo instalada la Máquina Virtual Java.

La única consideración es que esta librería esta compilada con gcc 3.2 así que si nuestro sistema no es compatible con esta versión de 3.2 tendremos que compilarla antes de copiarla.

Para que este plugin funcione al 100% es necesario tener instalado GEF (Graphical Editing Framework). Eclipse Profiler Plugin usa este otro plugin para dibujar los grafos de llamadas entre métodos, así que es muy recomendable tenerlo instalado. Lo podemos encontrar en: <http://www.eclipse.org/gef/>

3. Ejecución

Ahora cuando queramos analizar el rendimiento de nuestra aplicación tendremos que hacer Run → Run As → Profiler.

Se abrirá una nueva perspectiva (Profiler perspective), donde veremos el avance de nuestra aplicación. Una vez ha terminado la ejecución es en esta perspectiva donde extraeremos todo el jugo a este plugin.

Tweets por @adictosaltrabaj

 **adictosaltrabajo**
@adictosaltrabaj

No sólo de tutoriales vive el hombre.

[youtube.com/user/AutentiaM...](https://www.youtube.com/user/AutentiaM...) y/o su canal de podcast [ivoox.com/escuchar-auten...](https://www.ivoox.com/escuchar-auten...) y descubre más



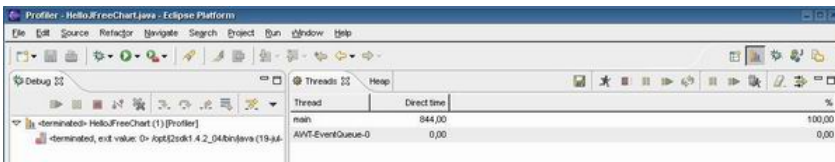
Insertar

Ver en Twitter

Archivos

Archivos

Elegir mes ▼

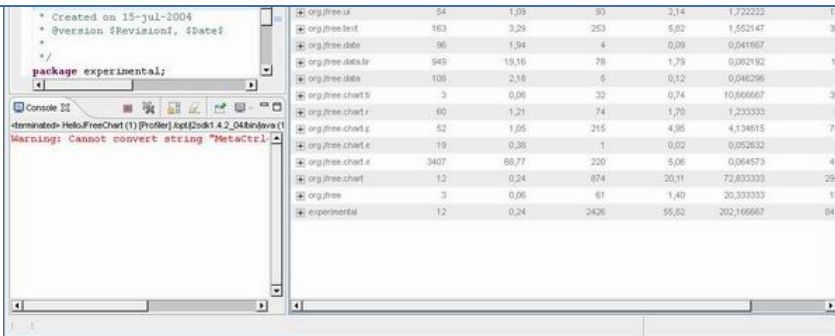


Thread	Direct time	%
main	844.00	100.00
AWT-EventQueue-0	0.00	0.00

Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información

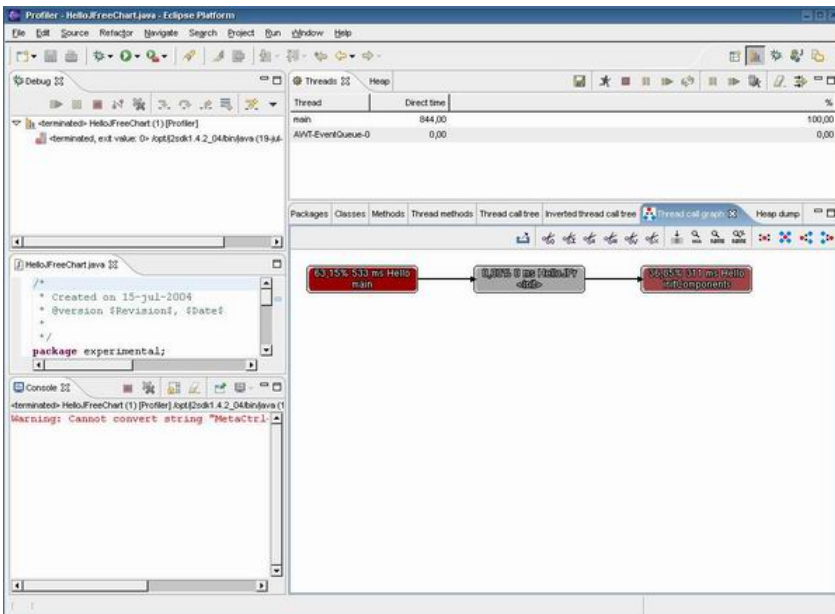
ACEPTAR



Package	Classes	Methods	Thread methods	Thread call tree	Inverted thread call tree	Heap dump
org.free.chart	54	1.09	93	2.14	1.722222	1.3
org.free.chart	163	3.29	253	5.82	1.552147	30
org.free.chart	96	1.94	4	0.09	0.041867	1
org.free.chart	949	19.16	78	1.79	0.002192	1
org.free.chart	108	2.18	5	0.12	0.048296	1
org.free.chart	3	0.06	32	0.74	10.866667	30
org.free.chart	60	1.21	74	1.70	1.233333	1
org.free.chart	52	1.05	215	4.95	4.134815	70
org.free.chart	19	0.38	1	0.02	0.052632	1
org.free.chart	3407	68.77	220	5.06	0.064573	40
org.free.chart	12	0.24	874	20.11	72.833333	290
org.free	3	0.06	61	1.40	20.333333	10
experimental	12	0.24	2426	55.82	202.166667	840

Como se ve en la imagen anterior podemos visualizar la información por paquete, clase, método, ... En cada una de estas vistas podemos obtener información sobre el número de llamadas, el tiempo por llamada, el tiempo total, ...

Una vista muy interesante es la de "Thread call graph", en esta vista se puede ver el grafo de llamadas entre los distintos métodos, y con un código de colores vemos los métodos que más tiempo han consumido (gris claro poco tiempo, rojo oscuro mucho tiempo). Para que se dibuje el grafo es necesario que seleccionemos (en la vista que esta justo encima) el thread que queremos representar.



Analizando los datos que pone a nuestra disposición esta perspectiva podemos localizar cual es ese 10% del código que resulta crítico.

Otra cosa que podemos hacer es exportar la información de esta perspectiva a html, de forma que una vez optimizado el código podemos volver a analizar y comparar, comprobando de esta forma que realmente ha mejorado el rendimiento de la aplicación.

4. Configuración

Una vez terminada la instalación podemos abrir Eclipse.

Si abrimos Window → Preferences → Profiler, veremos la pestaña de configuración de Eclipse Profiler Plugin.

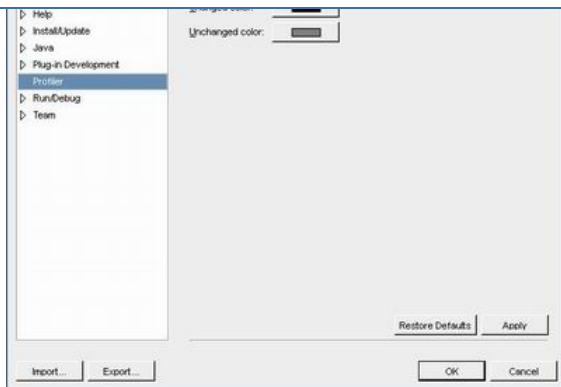
Como podéis ver en la imagen abajo, es muy sencilla, de hecho lo único que podemos configurar son los colores:

colores:

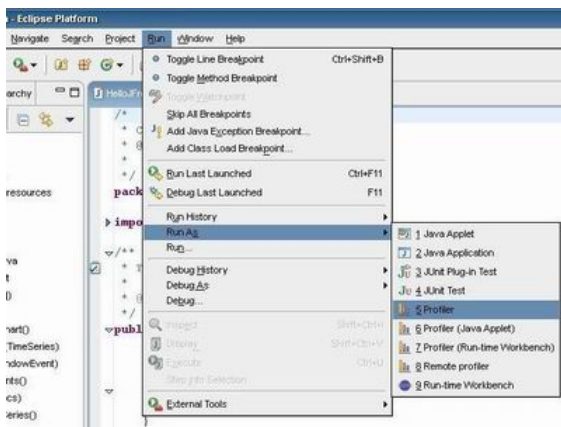
Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información

ACEPTAR



Como se ha comentado en el punto anterior, se han añadido nuevas opciones para ejecutar las aplicaciones desde el menú Run → Run As:



Si pinchamos sobre Run → Run... veremos que hay dos pestañas nuevas: Profiler y Profiler adv. Estas dos pestañas nos van a resultar de mucha utilidad para configurar el comportamiento del profiler durante la ejecución. Por ejemplo podemos marcar los paquetes que no queremos que se tengan en cuenta al hacer el profile, como las clases que no hemos desarrollado nosotros, y que normalmente no modificaremos (en muchos casos ni siquiera tenemos los fuentes de estas clases).

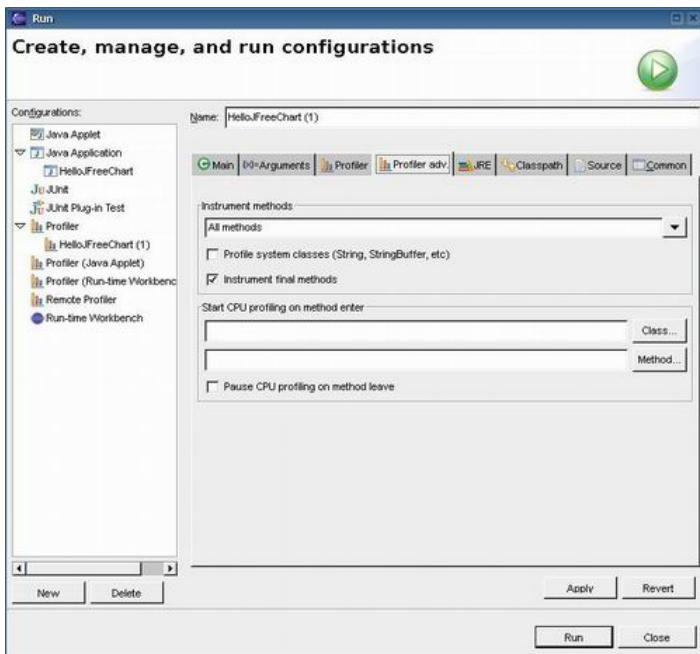
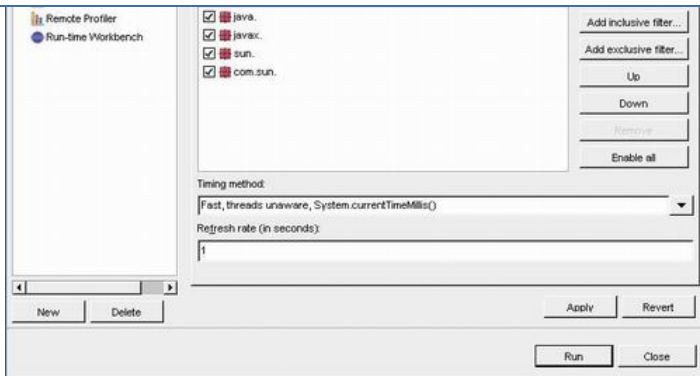
Otra cosa que puede resultar muy útil es indicar cuando debe empezar a contarse los tiempos. Indicando una clase y un método podemos hacer que en vez de contar los tiempos desde que se arranca la aplicación, se haga al llegar a la primera ejecución de ese método. Esto puede ser muy útil para saltarnos una primera etapa de inicialización que no queremos contabilizar.



Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información

ACEPTAR



5. Conclusiones

A la hora de escribir aplicaciones y mejorar el rendimiento podemos seguir estos puntos:

- No se debe escribir el código desde un punto de vista de optimización.
- Debemos centrarnos en un buen diseño y lógica de negocio.
- El código debe ser legible para favorecer la mantenibilidad (a la larga, ahorro de recursos).
- Uso de herramientas especializadas, como Eclipse Profiler Plugin.
- Centrar los esfuerzos de optimización en el 10% del código realmente crítico.

Siguiendo estos pequeños consejos, y con las herramientas apropiadas (como Eclipse Profiler Plugin) podemos llegar a un buen equilibrio entre código legible y mantenible, y una aplicación con un excelente rendimiento.

6. Sobre el autor

Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra política de cookies, pinche el enlace para mayor información

[ACEPTAR](#)

<mailto:alejandropg@autentia.com>

Autentia Real Business Solutions S.L.

<http://www.autentia.com>

Comparte este artículo!



Esta obra está licenciada bajo [licencia Creative Commons de Reconocimiento-No comercial-Sin obras derivadas 2.5](#)

Deja un comentario

Name (required)

E-Mail (required)

Website

Enviar comentario

Lo último

Arquitectura Serverless con
Lambdas sobre AWS
Terraform para provisionar en
AWS
Introducción a NestJS
Cómo eliminar el chroma de una
fotografía en menos de 5 minutos

Datos de contacto

Edificio BestPoint Avd. de Castilla,
1, Planta 2, Oficina 21B (San
Fernando de Henares)
Phone: 916 75 33 06
E-Mail:
adictos@adictosaltrabajo.com
Web: <https://www.autentia.com>

Powered by



Property-based testing con
ScalaCheck.

Uso de cookies

Este sitio web utiliza cookies para que usted tenga la mejor experiencia de usuario. Si continúa navegando está dando su consentimiento para la aceptación de las mencionadas cookies y la aceptación de nuestra [política de cookies](#), pinche el enlace para mayor información

[ACEPTAR](#)
