Eines de visualització del "Garbage Collector"

A qui va dirigit
Desenvolupadors d'aplicacions basades en Canigó.

Versió de Canigó
Utilitzable en qualsevol versió de Canigó.

Introducció
En gairebé qualsevol aplicació Java sempre es interessant conèixer l'evolució del Garbage Collector (GC) per a detectar possibles errors en la execució de la nostra aplicació.
A més, la configuració del GC afecta de forma directa al rendiment de les aplicacions que estiguin funcionant sobre una màquina virtual.

Garbage Collector

El Garbage Collector és una eina integrada dintre de cada màquina virtual de Java que s'encarrega de la neteja del HEAP, la memòria assignada a la màquina virtual. Bàsicament es tracta d'un conjunt de polítiques d'alliberament de memòria que s'executen de forma automàtica.

En el moment de l'execució del GC tots els *threads* que estiguin iniciats sobre la màquina virtual s'aturen. Aquest comportament pot afectar al rendiment de les aplicacions que els utilitzin, especialment si són aplicacions on és crític un temps de resposta baix.

Existeixen diverses maneres de configurar el GC de forma que la seva execució sigui el menys perjudicial possible sobre les aplicacions. La configuració ideal sempre depèn del tipus d'aplicació que es vulgui executar i l'ús que faci de la memòria (creació / destrucció de molts objectes, manipulació d'objectes grans, ...).

Configuració del log del Garbage Collector

De forma natural les traces del GC es troben desactivades en l'execució d'un procés Java. En el context de les aplicacions web, el procés Java que executa les aplicacions es el servidor d'aplicacions. A continuació oferim un exemple d'activació de traces del servidor d'aplicacions Tomcat utilitzant la màquina virtual de Sun.

Per poder indicar a la JVM que mostri les traces del GC es necessari:

- Aturar el Tomcat
- Incloure en la variable CATALINA_OPTS el següent contingut:

```
export CATALINA_OPTS="-XX:+PrintGCDetails -XX:+PrintGCTimeStamps -verbose:gc -Xloggc:/var/log/gc.log"
```

- Arrancar el Tomcat

En aquest moment cada cop que s'executi el GC per alliberar memòria deixarà una traça en "/var/log/gc.log".

Per altres servidors d'aplicacions o distribució de màquina virtual, la configuració es similar. Només cal modificar els paràmetres d'inici corresponents.

Eines de visualització del "Garbage Collector"

Gcviewer

Per poder explotar de forma efectiva la informació emmagatzemada en el log del GC existeixen diferents eines. En aquest document es farà referència a l'eina GCviewer. Aquesta eina és gratuïta i aporta la possibilitat d'explotar el log deixat pel GC. Per a més detall sobre la naturalesa de l'eina podeu consultar la pàgina web del creador http://www.tagtraum.com/gcviewer.html

On descarregar-la?

http://www.tagtraum.com/gcviewer-download.html

Execució Eina

L'eina es presenta com un jar, per exemple gcviewer-1.28.jar. Per tal de executar-la només cal fer:

java -jar gcviewer-1.28.jar

tant si és un sistema Windows com Unix. En arrencar apareixerà la següent finestra:

Figura 1. Aplicació Gcviewer

Utilització de l'eina

Per poder explotar les dades d'un GC és necessari indicar a l'eina sobre quin arxiu de log del GC es vol treballar. Això es pot fer des de el menú "File" -> "Open File".
Una vegada seleccionat l'arxiu de log del GC apareixerà una gràfica corresponent a l'evolució de les execucions del GC durant tot el temps que indiqui l'arxiu de log.

Eines de visualització del "Garbage Collector"

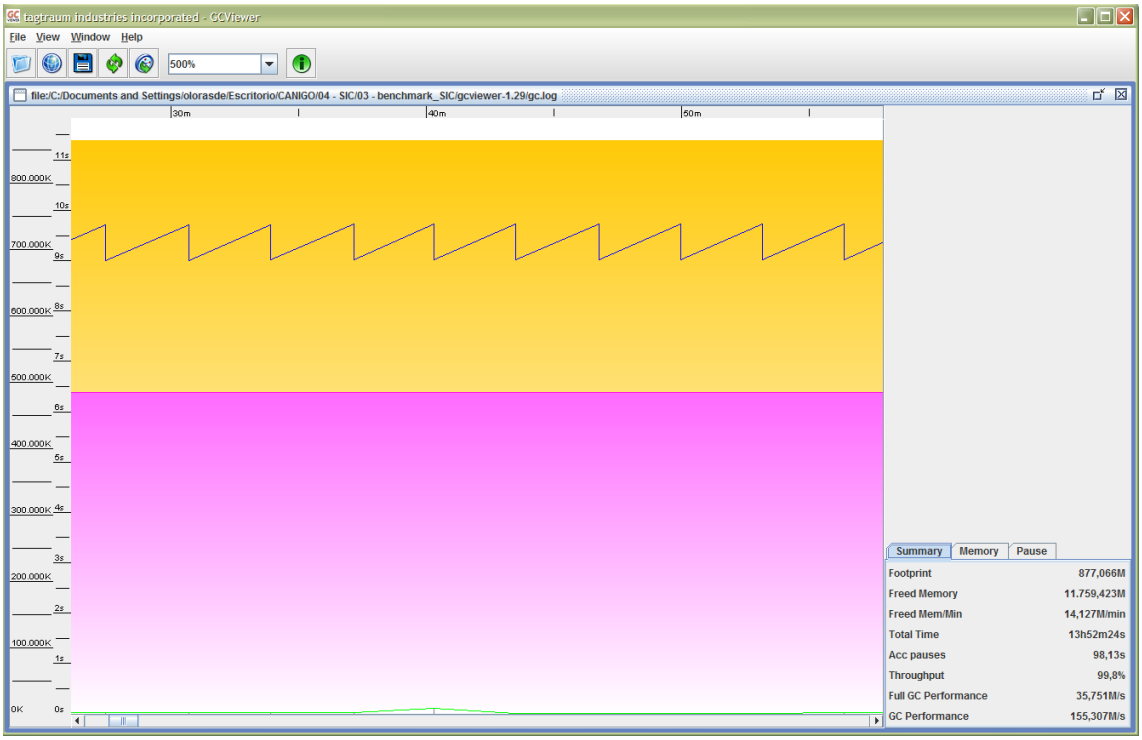

Figura 2. Execució d'un Minor Collection del GC

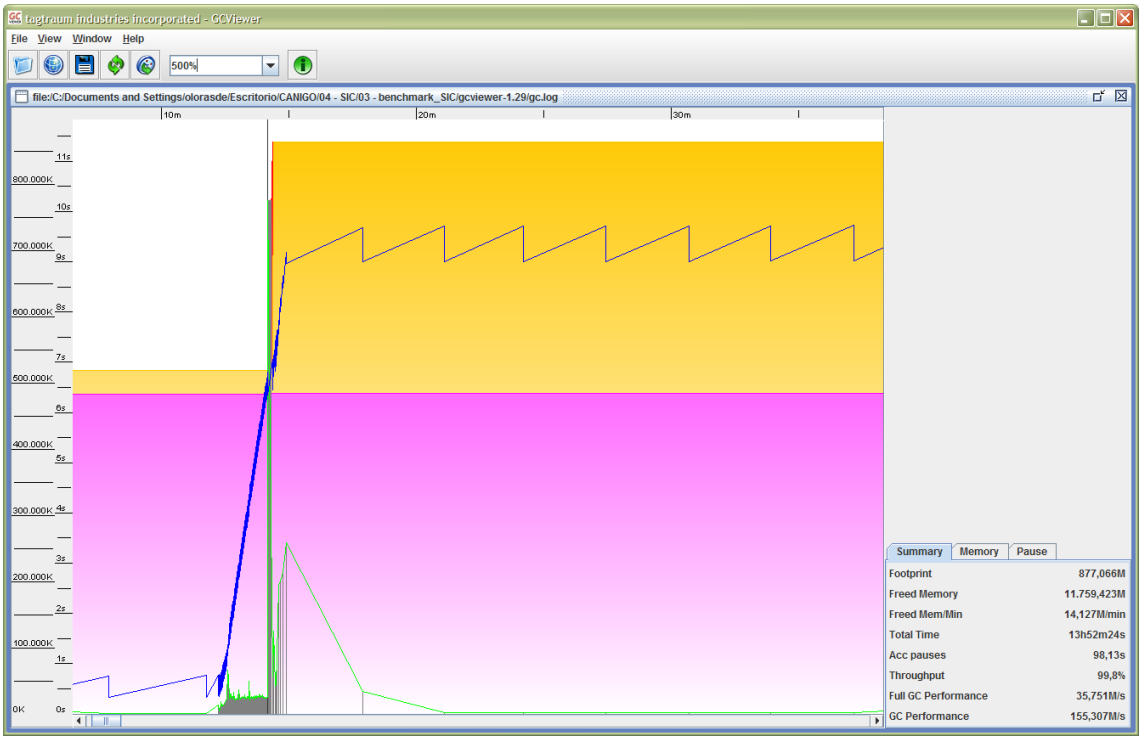

Figura 3. Execució d'un Major Collection del GC

En la gràfica que es mostra es poden observar una sèrie de paràmetres. De entre tots ells destaquem:

- L'evolució del consum de memòria del HEAP, representat per la línia de color blau fosc.

Eines de visualització del "Garbage Collector"

- El temps i la durada de l'execució del GC representat per la línia de color verd.
- El footprint, la quantitat total de memòria gastada per l'aplicació en el transcurs de la seva execució.
- L'espai de Young generation del HEAP, representat amb color de fons taronja.
- L'espai de Tenured generation del HEAP, representat amb color de fons lila.

Introducció funcionament de la memòria virtual (HEAP)

Per poder entendre els resultats de la gràfica s'ha de conèixer el funcionament del GC. Inicialment existeixen dues zones de memòria que composen el HEAP. La zona de nova generació (Young Generation, YG) i la zona d'emmagatzemament d'objectes antics (Telured Generation TG).

En iniciar una aplicació tots els nous objectes són situats en la YG. Arriba un moment en que aquesta zona de memòria creix tant que el GC ha de començar a moure objectes de la YG a la TG. Aquest procés s'anomena Minor Collection. És molt ràpid i consumeix pocs recursos. Es pot apreciar aquest tipus de execució en la gràfica del GCviewer com a petites variacions del HEAP que simulen una figura de "serra" veure figura 2.

Fent aquest procés arribarà un moment en que els objectes que hi ha a la YG no hi cabran en la TG. En aquest moment s'executa un procés de Mayor Collection en la zona de memòria TG per agrupar i compactar els objectes emmagatzemats allà per poder deixar espai als nous objectes provinents de la zona YG.

És precisament aquest procés el que resulta més costos per a la màquina virtual i el que congela tots els threads fins la seva finalització.

Interpretació de l'evolució de la memòria

En la figura 3 es pot apreciar l'execució d'un procés de Mayor Collection. En el cas de l'exemple el temps en que la màquina virtual **ha estat "congelada" són 10 segons**.

En condicions normals l'espai de YG acostuma a ser 1/3 del total reservat. Aquest paràmetre és variable. Una YG més gran farà que els Minor Collections siguin una mica més lents però allargarà el temps abans de que s'executi un Major Collection.

Un aspecte important que també es pot detectar amb aquesta eina és la presencia de *memory leaks*, o pèrdues de memòria. Si en una gràfica del GC s'observa un conjunt de Minors Collections amb un tendència sempre ascendent pot indicar la presència d'aquest efecte. En el moment que no quedi espai disponible al HEAP, es produirà un error d'execució en l'aplicació informant que no queda memòria disponible.

Per més informació d'aspectes de *tuning* i funcionament del GC es pot consultar el següent enllaç:

- SUN: http://java.sun.com/docs/hotspot/gc5.0/gc_tuning_5.html