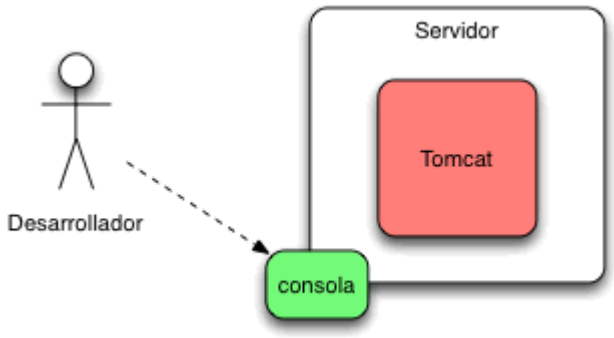
JavoSize es una herramienta de troubleshooting que permite localizar y solventar muchos de los problemas que nos puedan aparecer en la puesta en producción de nuestras aplicaciones Java . Si algo destaca de esta herramienta es que no hace falta ser un guru para manejarse con ella, algo que lo diferencia de otras.

Configuración

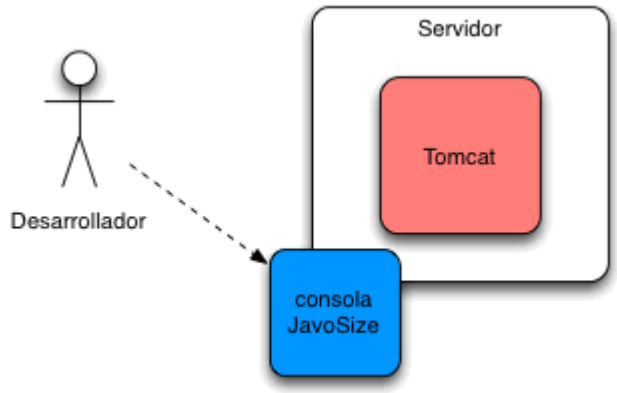
Normalmente cuando tenemos problemas con una aplicación , siempre estamos delante de la consola del servidor que la tiene instalada ya sea de forma local o remota.



JavoSize se integra de forma sencilla con esta filosofía ya que permite que una vez tengamos el servidor arrancado conectarnos a él a través de una consola “potenciada”. Para ello únicamente se necesita descargar el jar de JavOSize y saber el PID del servidor, desde la consola ejecutamos.

java -jar javosize-1.1.3.jar 707

Donde 707 es el PID del servidor. Esto nos dará acceso a una nueva consola interna en donde podemos tener un control a más detalle de lo que esta ocurriendo.



Esta consola funciona de una forma muy similar a la consola batch habitual pero esta especializada en el mundo Java. Podemos por ejemplo ejecutar el comando ls.

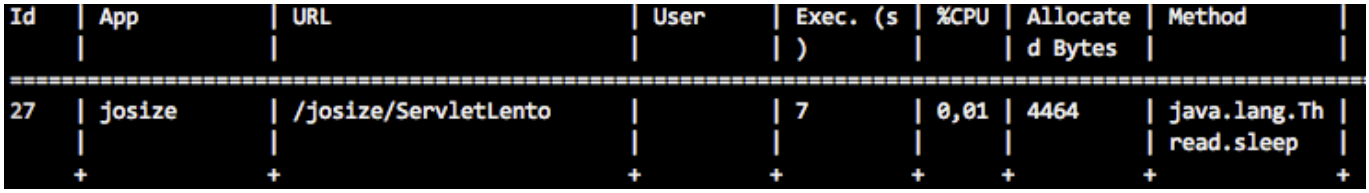
```
[javOSize@JVM /]~> ls
apps
appthreads
breakpoints
classes
custommetrics
interceptor
jmx
memory
perfcounter
problems
repository
scheduler
sessions
sh
threads
users
```

Este comando nos devolverá las diferentes opciones que existen de la herramienta. Vamos a ver un par de casos de uso típicos. Partimos de que el proceso que hemos arrancado es un Tomcat 8 y que este dispone de un Servlet.

```
package com.arquitecturajava;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
* Servlet implementation class ServletLento
*/
@WebServlet("/ServletLento")
public class ServletLento extends HttpServlet {
private static final long serialVersionUID = 1L;

protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {

try {
Thread.sleep(10000);
} catch (InterruptedException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}

PrintWriter pw= response.getWriter();
pw.println("que tento va");
```

```
}

}
```

Como se puede observar el Servlet tarde 10 segundos en ejecutarse ya que usamos Thread.sleep(10000) . La herramienta podrá mostrárnoslo en tiempo real con solo acceder a la carpeta de appThreads con cd y usar el comando ls.



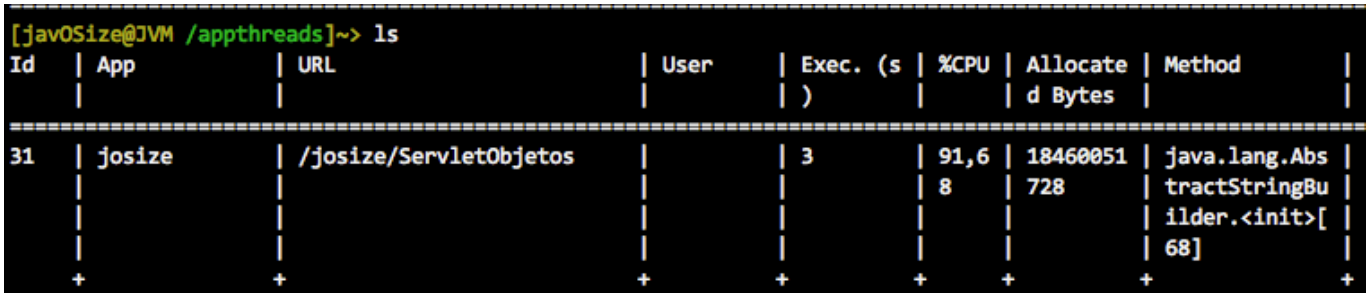
Id	App	URL	User	Exec. (s)	%CPU	Allocated Bytes	Method
27	josize	/josize/ServletLento		7	0,01	4464	java.lang.Thread.sleep

Se puede observar que el Servlet lleva 7 segundos ejecutandose con el problema que esto conlleva. ¿Qué pasaría si en vez de ejecutar ese Servlet ejecutamos este otro?

```
package com.arquitecturajava;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
```

```
/**
* Servlet implementation class ServletObjetos
*/
@WebServlet("/ServletObjetos")
public class ServletObjetos extends HttpServlet {
private static final long serialVersionUID = 1L;

protected void doGet(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {

String cadena="";
for (int i = 0; i < 20000; i++) {

cadena = cadena+"hola" + i +"que" + "tal"+i;
}

PrintWriter pw = response.getWriter();
pw.println("objetos");

}

}
```

En este caso estamos obligando a Java a construir miles de cadenas con el uso de CPU que esto implica. Si volvemos a lanzar un comando ls con la herramienta podremos ver de forma rápida el consumo.



En este caso este Servlet consume el 91% de CPU es evidente que es ahí donde tenemos el principal problema.

Otros artículos relacionados: Java Strings , StringBuilder Rendimiento