Un poco de Java y +

Otra forma de hablar de nuestro día a día…

9 MARZO 2020
LUISMI GRACIA

Un poco de Byteman: Primeros Pasos

(https://unpocodejava.files.wordpress.com/2020/03/image003.jpg)

Byteman (https://byteman.jboss.org/) es un herramienta creada por la comunidad JBoss.org que nos ayuda a **tracear, monitorizar y testar el comportamiento de una aplicción Java.**

Para esto inyecta código Java en los métodos Java sin la necesidad de recompilar, reempaquetar o redesplegar la aplicación. La inyección puede hacerse en el arranque de la JVM o incluso mientras la aplicación está ejecutando. Además el código inyectado puede acceder a los datos y llamar a otros métodos (includo métodos privados).

Byteman funciona modificando el bytecode de las clases de la aplicación en runtime. A diferencia de otras librerías Byteman opera a nivel de Java, definiendo las reglas que especifican el código a modificar. Byteman es capaz de modificar el código de cualquier librería, incluso de las librerías del runtime de la JVM.

Dicho esto, **seguro que le encontráis numerosos usos a Byteman, ¿verdad?**

El uso más simple de Byteman es inyectar sentencias de log para lo que está haciendo la aplicación, mostrar el valor de ciertos datos,…de modo que se pueda monitorizar y debugear nuestras aplicaciones en ejecución. Además usar Byteman tiene la ventaja de no afectar al rendimiento al no tener que modificar el código para poner trazas.

Veamos cómo se usa

1. Comenzaremos por descargar el binario de Byteman (Byteman 4.0.11 (https://byteman.jboss.org/downloads.html)) que es un ZIP y descomprimirlo en una carpeta.
2. Luego establecemos una variable BYTEMAN_HOME apuntando a esta carpeta y la añadimos al %PATH% el %BYTEMAN_HOME%/bin
3. Ya podemos probarlo, primero un ejemplo sencillo, escribimos una clase sencillita:

```
public class BytemanFirstExample {

    public static void main(String[] args) {
        for (int i = 0; i < args.length; i++) {
            System.out.println(args[i]);
        }
    }
}
```

(https://unpocodejava.files.wordpress.com/2020/03/image008-1.jpg)

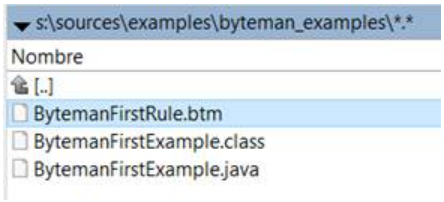
4. La compilamos y la ejecutamos

```
s:\sources\examples\byteman_examples>javac BytemanFirstExample.java

s:\sources\examples\byteman_examples>java BytemanFirstExample uno dos tres
uno
dos
tres
```

(https://unpocodejava.files.wordpress.com/2020/03/image009-1.jpg)

5. Ahora nos queda escribir la RULE Byteman, la llamaré **BytemanFirstRule.btm** y La dejaré en la misma carpeta



(https://unpocodejava.files.wordpress.com/2020/03/image011-1.jpg)

6. La regla tendrá este aspecto (la pongo tabulada para que se lea mejor pero no es necesario):

```
RULE trace main entry
    CLASS BytemanFirstExample
        METHOD main
            AT ENTRY
                IF true
                    DO traceln("---> Empieza el main...")
ENDRULE

RULE trace main exit
    CLASS BytemanFirstExample
        METHOD main
            AT EXIT
                IF true
                    DO traceln("---> Acaba el main")
ENDRULE
```

(https://unpocodejava.files.wordpress.com/2020/03/image013.jpg)

Como se puede ver la regla tiene 2 reglas, una que se ejecuta a la entrada del método main y otra a la salida:

7. Ahora ya puedo ejecutarla, simplemente añadiendo el javaagent y la ruta al script:

java -javaagent**:%BYTEMAN_HOME%/lib/byteman.jar=script:BytemanFirstRule.btm** BytemanFirstExample uno dos tres

```
s:\sources\examples\byteman_examples>java -javaagent:%BYTEMAN_HOME%/lib/byteman.jar=script:BytemanFirstRule.btm BytemanFirstExample uno dos tres
---> Empieza el main...
uno
dos
tres
---> Acaba el main
```

(https://unpocodejava.files.wordpress.com/2020/03/image015.jpg)

Esto está bien, ¿pero es todo?

Noooooooo!!! Veamos un caso más complejo.

Primero la clase:

```java
public class BytemanThreadExample {
    public static void main(String[] args)     {

        for (int i = 0; i < args.length; i++) {
            final String arg = args[i];
            Thread thread = new Thread(arg) {
                public void run() {
                System.out.println(arg);
                }
            };
            thread.start();
            try {
                thread.join();
            } catch (Exception e) {
            }
        }
    }
}
```

(https://unpocodejava.files.wordpress.com/2020/03/image021.jpg)

Y mi fichero btm

```
RULE trace thread start
    CLASS java.lang.Thread
        METHOD start()
            IF true
            DO traceln("*** Comienza el Hilo: "+ $0.getName())
ENDRULE
```

(https://unpocodejava.files.wordpress.com/2020/03/image023.jpg)

En este caso quiero inyectar código en la clase Thread así que la forma de lanzarlo es un poco más compleja:

java -javaagent:%**BYTEMAN_HOME%libbyteman.jar=script:BytemanThreadRule.btm,boot:%BYTEMAN_HOME%libbyteman.jar -Dorg.jboss.byteman.transform.all** BytemanThreadExample Hilo1 Hilo2 Hilo3

```
s:\sources\examples\byteman_examples>java -javaagent:%BYTEMAN_HOME%\lib\byteman.jar=script:BytemanThreadRule.btm,boot:%BYTEMAN_HOME%\lib\byteman.jar -Dorg.jboss.byteman.trans
form.all BytemanThreadExample Hilo1 Hilo2 Hilo3
*** Comienza el Hilo: Hilo1
Hilo1
*** Comienza el Hilo: Hilo2
Hilo2
*** Comienza el Hilo: Hilo3
Hilo3
```

(https://unpocodejava.files.wordpress.com/2020/03/image026.jpg)

En el próximo post (este miércoles o jueves si el CoronaVirus no lo impide!!!) veremos cómo cargar y descargar Reglas sobre un programa en ejecución! 😉

JAVA **OPEN SOURCE** **QUÉ ES** **UN POCO DE**