

PRÁCTICA 03 2DAM 17/11/2023

Multiprocesos

Manuel Martín Santamaría

Índice

- ❖ [Pag 3] - [Indicaciones de entrega](#)
- ❖ [Pag 4] - [Actividades](#)
 - [Pag 4] - [Actividad 1](#)
 - [Pag 4] - [Actividad 2](#)
 - [Pag 4] - [Actividad 3](#)
 - [Pag 4] - [Actividad 4](#)
 - [Pag 4] - [Actividad 5](#)
 - [Pag 5] - [Actividad 6](#)
 - [Pag 6] - [Actividad 9](#)
 - [Pag 6] - [Actividad 10](#)
 - [Pag 6] - [Actividad 11](#)

Indicaciones de entrega

Una vez realizada la tarea elaborarás un único fichero .zip donde figuren las respuestas correspondientes. El envío se realizará a través de la plataforma de la forma establecida para ello, y el archivo se nombrará siguiendo las siguientes pautas:

apellido1_apellido2_nombre_SIGxx_Tarea

Asegúrate que el nombre no contenga la letra ñ, tildes ni caracteres especiales extraños. Así por ejemplo la alumna Begoña Sánchez Mañas para la primera unidad del MP de DAW, debería nombrar esta tarea como...

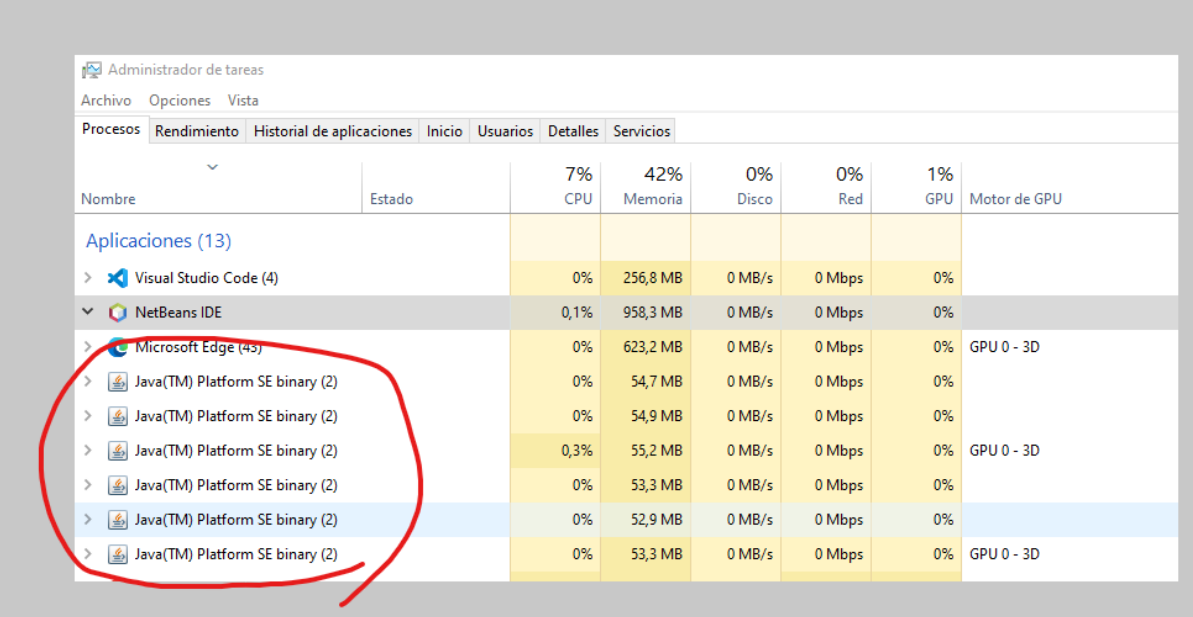
sanchez_manas_begona_PSP_Tarea01

Cada ejercicio ha de desarrollarse en un documento distinto, siguiendo los nombres de ej1, ej2,...

Actividades

[Actividad 1]

Visualiza los procesos con el administrador identificando la información que consideres relevante. Observa que cada vez que se pulsa un botón, se abre un proceso nuevo (sin límite).



Nombre	Estado	7% CPU	42% Memoria	0% Disco	0% Red	1% GPU	Motor de GPU
Aplicaciones (13)							
Visual Studio Code (4)		0%	256,8 MB	0 MB/s	0 Mbps	0%	
NetBeans IDE		0,1%	958,3 MB	0 MB/s	0 Mbps	0%	
Microsoft Edge (43)		0%	623,2 MB	0 MB/s	0 Mbps	0%	GPU 0 - 3D
Java(TM) Platform SE binary (2)		0%	54,7 MB	0 MB/s	0 Mbps	0%	
Java(TM) Platform SE binary (2)		0%	54,9 MB	0 MB/s	0 Mbps	0%	
Java(TM) Platform SE binary (2)		0,3%	55,2 MB	0 MB/s	0 Mbps	0%	GPU 0 - 3D
Java(TM) Platform SE binary (2)		0%	53,3 MB	0 MB/s	0 Mbps	0%	
Java(TM) Platform SE binary (2)		0%	52,9 MB	0 MB/s	0 Mbps	0%	
Java(TM) Platform SE binary (2)		0%	53,3 MB	0 MB/s	0 Mbps	0%	GPU 0 - 3D

[Actividad 2]

a. ¿Qué ocurre?

Si ejecuto el programa desde NetBeans (IDE), la salida del proceso se mostrará en la consola del IDE.

b. ¿Qué puede estar pasando?

Si ejecuto el programa desde cmd, la salida aparece directamente en la consola.

c. Investiga cómo poder visualizar la información desde el proceso anfitrión.

Para visualizar la información desde el proceso anfitrión, se puede redirigir la salida del proceso y luego leerla desde el programa Java utilizando `ProcessBuilder.redirectOutput`. Eso sí, si el comando requiere de permisos elevados y el programa no los tiene, puede fallar.

[Actividad 3]

¿Qué sucede si el comando insertado no existe o está mal escrito? ¿lo notifica de alguna forma?

Cuando el comando insertado no existe o está mal escrito, se captura la excepción `IOException` que se lanza cuando se intenta ejecutar el proceso

[Actividad 4]

¿Es multiplataforma el programa actual?

No es completamente multiplataforma (aunque java sea un lenguaje compilado interpretable en múltiples plataformas si cuentan con la máquina virtual de java) debido a que algunos de los comandos y aplicaciones que utiliza son específicos de Windows.

En el caso del botón 2 y 3. ¿Habría que hacer algún cambio?

En el tercero en teoría no, porque el comando para ejecutar la aplicación java no varía, y yo lo tengo como path relativo, no influye el separador del sistema.

En el segundo, sin embargo, la forma de llamar a la ejecución de un programa sí varía de linux a windows.

[Actividad 5]

En C, existe un método para recoger el identificador de un proceso (`getppid(void);`) ¿existe algún método parecido en Java? ¿Cuál es?

En Java, no hay un método como `getppid` para obtener directamente el identificador del proceso.

No obstante, es posible obtener el identificador del proceso actual (PID) utilizando `ManagementFactory.getRuntimeMXBean().getName()`, esto devuelve una cadena con formato "pid@hostname", dividiendo esa cadena mediante `.split("@")` se puede extraer el PID.

[Actividad 6]

Mata uno de los procesos y lo notifícalo por pantalla. ¿Cuál mata?

Por como lo he codificado el ejercicio, solo mata al proceso que decido matar, sin embargo, dependiendo de la relación jerárquica de los procesos, en otro caso, si matas un proceso, puede conllevar el cierre de los demás si estos están fuertemente relacionados, ej: mato al proceso padre, se cierran sus procesos hijos.

[Actividad 9]

¿Qué sucede si se mata a un padre? ¿también matan a los hijos o si estos se quedan activos?

Al matar a un proceso padre, como sus procesos hijos heredan el grupo de procesos del padre, cuando se termina el proceso padre, se envía una señal de terminación a todo el grupo de procesos, incluidos los hijos.

Si destruyo el proceso abuelo, que es el proceso actual, lo cual no está permitido, salta una excepción.

```
run:
Soy el proceso ABUELO 9132
Exception in thread "AWT-EventQueue-0" java.lang.IllegalStateException: destroy of current process not allowed
    at java.base/java.lang.ProcessHandleImpl.destroyProcess(ProcessHandleImpl.java:369)
    at java.base/java.lang.ProcessHandleImpl.destroyForcibly(ProcessHandleImpl.java:392)
    at psp_ud02_act1.ej9.Ej9.main(Ej9.java:30)
    at psp_ud02_act1.Indice jButton9ActionPerformed(Indice.java:194)
    at psp_ud02_act1.Indice$9.actionPerformed(Indice.java:127)
    at java.desktop/javax.swing.AbstractButton.fireActionPerformed(AbstractButton.java:1972)
    at java.desktop/javax.swing.AbstractButton$Handler.actionPerformed(AbstractButton.java:2314)
    at java.desktop/javax.swing.DefaultButtonModel.fireActionPerformed(DefaultButtonModel.java:407)
    at java.desktop/javax.swing.DefaultButtonModel.setPressed(DefaultButtonModel.java:262)
    at java.desktop/javax.swing.plaf.basic.BasicButtonListener.mouseReleased(BasicButtonListener.java:279)
    at java.desktop/java.awt.Component.processMouseEvent(Component.java:6620)
```

Si destruyo al padre desde el abuelo, como más adelante espero a la terminación de la ejecución del proceso padre, el proceso se queda colgado esperando indefinidamente la respuesta del hijo que hemos matado (que no llegará).

```
ProcessHandle abueloHandle = ProcessHandle.current();

long miPid = abueloHandle.pid();
long padrePid;

System.out.println("Soy el proceso ABUELO " + miPid);

Process padre = iniciarPadre(miPid);

// Matar el proceso padre
padre.destroy();

// Obtener la salida del proceso
InputStream inputStream = padre.getInputStream();
BufferedReader reader = new BufferedReader(new InputStreamReader(i

padrePid = padre.pid();

try {
    padre.waitFor();
} catch (InterruptedException ex) {
    ex.printStackTrace();
}
```

[Actividad 10]

¿Puede realizarse en una sola instrucción de línea de comandos? ¿Qué habría que hacer?

Si, se puede redirigir la salida de un comando a otro y luego a un fichero.

Se puede usar el comando tasklist en Windows junto pasando mediante una tubería la salida al comando findstr para filtrar los resultados, después se puede redirigir la salida a un archivo con ">".

[Actividad 11]

¿Qué sucede si está mal alguna de esas instrucciones? Haz que muestre el correcto funcionamiento y también el error en caso de fallo.

Si alguna de las instrucciones está mal, no sale por el flujo de la salida estándar del proceso, si no por la de error.

He obtenido el flujo de salida estándar y el de error por separado, y los leo en bucle en caso de que el proceso devuelva uno u otro, al este generar líneas de salida, el bucle correspondiente los leerá.

```
// Capturar la salida estándar
InputStream inputStream = proceso.getInputStream();
BufferedReader inputReader = new BufferedReader(new InputStreamReader(in: inputStream));

// Capturar la salida de error
InputStream errorStream = proceso.getErrorStream();
BufferedReader errorReader = new BufferedReader(new InputStreamReader(in: errorStream));

// Leer y mostrar la salida estándar
String line;
while ((line = inputReader.readLine()) != null) {
    System.out.println("Salida estándar: " + line);
}

// Leer y mostrar la salida de error
while ((line = errorReader.readLine()) != null) {
    System.err.println("Salida de error: " + line);
}
```

Posteriormente, en caso de éxito, se imprime el mensaje:

"Instrucción ejecutada con éxito: " + instruccion

En caso de error el mensaje:

"Error al ejecutar la instrucción: " + instruccion

```
// Esperar a que el proceso termine
int exitCode = proceso.waitFor();

if (exitCode == 0) {
    System.out.println("Instrucción ejecutada con éxito: " + instruccion);
} else {
    System.err.println("Error al ejecutar la instrucción: " + instruccion);
}

} catch (IOException | InterruptedException e) {
    System.err.println("Error al ejecutar la instrucción: " + instruccion);
    e.printStackTrace();
}
```


Teniendo lo anterior en cuenta, al observar la salida del programa, me surge alguna duda...

- En la salida de “cmd /c echo Hola Mundo” se aprecia de color blanco y mensaje de éxito.
- En la salida de “mondongo” un comando falso para forzar el error, se aprecia de color rojo y el mensaje de error.
- En la salida de “cmd /c dir” se aprecia de color blanco y mensaje de éxito.

Hasta aquí todo bien, pero en el último:

- En la salida de “java -version” , se aprecia de color rojo, precedido por “salida de error”, lo que indica que ha salido por el flujo de salida de error del proceso, como si fuese un error, pero el mensaje de éxito, lo que indica que el exit code ha sido 0 (de éxito).

```

Run:
Archivo de prueba generado correctamente.
Salida estándar: Hola Mundo
Instrucción ejecutada con éxito: cmd /c echo Hola Mundo
Error al ejecutar la instrucción: mondongo
java.io.IOException: Cannot run program "mondongo": CreateProcess error=2, El sistema no puede encontrar el archivo especificado
Caused by: java.io.IOException: CreateProcess error=2, El sistema no puede encontrar el archivo especificado
    at java.base/java.lang.ProcessImpl.create(Native Method)
    at java.base/java.lang.ProcessImpl.<init>(ProcessImpl.java:500)
    at java.base/java.lang.ProcessImpl.start(ProcessImpl.java:159)
    at java.base/java.lang.ProcessBuilder.start(ProcessBuilder.java:1111)
    ... 40 more
Salida estándar: El volumen de la unidad D es HDD (Datos)
Salida estándar: El número de serie del volumen es: 7AAD-6956
Salida estándar:
Salida estándar: Directorio de D:\Proyectos\2SDam\DAM2_ProgramacionDeServiciosYProcesos\Tema2\PSP_UD02_ACT1
Salida estándar:
Salida estándar: 17/11/2023 08:14 <DIR> .
Salida estándar: 17/11/2023 08:14 <DIR> ..
Salida estándar: 17/11/2023 08:14 <DIR> build
Salida estándar: 15/11/2023 04:25 3.624 build.xml
Salida estándar: 17/11/2023 08:14 <DIR> dist
Salida estándar: 17/11/2023 08:14 61 instrucciones.txt
Salida estándar: 15/11/2023 04:25 85 manifest.mf
Salida estándar: 15/11/2023 04:41 <DIR> nbproject
Salida estándar: 15/11/2023 04:25 5.615 Output.jar
Salida estándar: 15/11/2023 05:43 994 outputEj3.txt
Salida estándar: 17/11/2023 07:39 1.921 procesoNieto.jar
Salida estándar: 17/11/2023 07:39 3.360 procesoPadre.jar
Salida estándar: 15/11/2023 04:25 <DIR> src
Salida estándar: 17/11/2023 07:55 6.806 SVCHOST.TXT
Salida estándar: 15/11/2023 04:52 <DIR> test
Salida estándar: 8 archivos 22.466 bytes
Salida estándar: 7 dirs 458.882.408.448 bytes libres
Instrucción ejecutada con éxito: cmd /c dir
Salida de error: java version "20.0.2" 2023-07-18
Salida de error: Java(TM) SE Runtime Environment (build 20.0.2+9-78)
Salida de error: Java HotSpot(TM) 64-Bit Server VM (build 20.0.2+9-78, mixed mode, sharing)
Instrucción ejecutada con éxito: java -version
  
```