PSP - U3 Actividades

Descargar estos apuntes

Índice

Nombre del proyecto

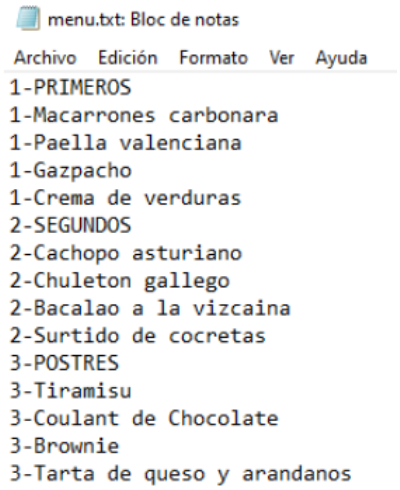
En estas actividades volvemos a usar la nomenclatura de clases y proyectos explicada en la hoja de actividades de autoevaluación de la U2

10. Clase psp.actividades.U3A10_Menu

Vamos a programar en Java la lectura del menú de un restaurante (menu.txt) que ofrece varios primeros, varios segundos y varios postres. Para ello, tendremos 3 hilos que formarán parte de un grupo de hilos (Threadrgoup) llamado “Menu diario” y que trabajarán del siguiente modo:

- Un hilo que lee de menu.txt solo los primeros (que vienen precedidos de la marca 1-) y los vuelca al archivo primeros.txt
- Un hilo que lee de menu.txt solo los segundos (que vienen precedidos de la marca 2-) y los vuelca al archivo segundos.txt
- Un hilo que lee de menu.txt solo los postres (que vienen precedidos de la marca 3-) y los vuelca al archivo postres.txt

Antes de codificar la solución en Java, tómate un tiempo en diseñar tu propio menú que debería tener el siguiente formato:



El objetivo es que cada hilo trabaje solo una parte del menú y que acabe escribiendo en su fichero pero sin el rótulo correspondiente y sin el número y el guión que precede a cada plato.
Es decir, en el fichero “primeros.txt” solo debería aparecer las siguientes lineas:

> Macarrones carbonara
> Paella valenciana
> Gazpacho
> Crema de verduras

Además, el programa imprimirá en cada momento el hilo que está actuando y qué está escribiendo. Un ejemplo de ejecución sería el siguiente, que utiliza el menú.txt de arriba:
En la consola, al finalizar el programa, se mostrarán las siguientes acciones:

```
El hilo 1 escribe... Macarrones carbonara
El hilo 1 escribe... Paella valenciana
El hilo 1 escribe... Gazpacho
El hilo 2 escribe... Cachopo asturiano
El hilo 1 escribe... Crema de verduras
El hilo 2 escribe... Chuletón gallego
El hilo 3 escribe... Tiramisú
El hilo 2 escribe... Bacalao a la vizcaina
El hilo 2 escribe... Surtido de croquetas
El hilo 3 escribe... Coulant de chocolate
El hilo 3 escribe... Brownie
El hilo 3 escribe... Tarta de queso y arándanos
```

Que habrá generado los siguientes ficheros:

primeros.txt	**segundos.txt**	**postres.txt**
Macarrones carbonara	Cachopo asturiano	Tiramisú
Paella valenciana	Chuletón gallego	Coulant de chocolate

primeros.txt	segundos.txt	postres.txt
Gazpacho	Bacalao a la vizcaina	Brownie
Crema de verduras	Surtido de croquetas	Tarta de queso y arándanos

11. Clase psp.actividades.U3A11_Vector

Vamos a programar en Java la gestión de 2 hilos que accederán por turnos a un vector de 10 posiciones para ir completándolo con valores aleatorios.

Tendremos un primer hilo que comenzará a completarlo de izquierda a derecha (hilo1) y otro segundo hilo que lo completará de derecha a izquierda (hilo2). Ambos lo irán informando con valores aleatorios cuyos límites inferior y superior se solicitarán al principio del programa.

Cada vez que uno de los hilos inserte un valor, esperaremos un tiempo para de esa forma ir viendo en tiempo real como se va informando el vector.

Se mostrarán las 10 posiciones del vector, que inicialmente estarán informadas a 0 y que poco a poco se irán informando hasta que todo el vector esté completo. Cuando un hilo complete una posición, lo marcará con su identificador (1 si lo ha informado el hilo1 o 2 si lo ha informado el hilo2).

Un ejemplo de ejecución sería el siguiente:

```
Introduzca limite inferior:
10
Introduzca limite superior:
500
0       1       2       3       4       5       6       7       8       9
-------------------------------------------------------------------------
288     0       0       0       0       0       0       0       0       0
1

0       1       2       3       4       5       6       7       8       9
-------------------------------------------------------------------------
288     0       0       0       0       0       0       0       0       464
1                                                                       2

0       1       2       3       4       5       6       7       8       9
-------------------------------------------------------------------------
288     116     0       0       0       0       0       0       0       464
1       1                                                               2

0       1       2       3       4       5       6       7       8       9
-------------------------------------------------------------------------
288     116     0       0       0       0       0       0       493     464
1       1                                                       2       2

0       1       2       3       4       5       6       7       8       9
-------------------------------------------------------------------------
288     116     251     0       0       0       0       0       493     464
1       1       1                                               2       2

0       1       2       3       4       5       6       7       8       9
-------------------------------------------------------------------------
288     116     251     0       0       0       0       103     493     464
1       1       1                                       2       2       2

0       1       2       3       4       5       6       7       8       9
-------------------------------------------------------------------------
288     116     251     165     0       0       0       103     493     464
1       1       1       1                               2       2       2

0       1       2       3       4       5       6       7       8       9
-------------------------------------------------------------------------
288     116     251     165     0       0       360     103     493     464
1       1       1       1                       2       2       2       2

0       1       2       3       4       5       6       7       8       9
-------------------------------------------------------------------------
288     116     251     165     224     0       360     103     493     464
1       1       1       1       1               2       2       2       2

0       1       2       3       4       5       6       7       8       9
-------------------------------------------------------------------------
288     116     251     165     224     433     360     103     493     464
1       1       1       1       1       2       2       2       2       2
```

12. Clase psp.actividades.U3A12_Controlador

> **Tip**
>
> La actividad 12 se hace a partir de la actividad `psp.actividades.U2A14_Controlador` que os he dejado como solución del tema 2.
>
> Debéis ver qué cambios son necesarios para cambiar el funcionamiento y pasar del uso de procesos al uso de hilos y adaptarla a la nueva actividad.

Haz un programa que calcule

$$\sum_{i=1}^{n} \sqrt{i}$$

El programa controlador recibe como parámetro el valor de n y, de forma opcional, también se le puede pasar la cantidad de operaciones a realizar por cada hilo (por defecto, si no se indica nada, 100).

Va a dividir equitativamente el reparto entre (n/100)+1 hilos.

- Al primer hilo le asignará el cálculo desde 0 hasta 99,
- al 2º desde 100 hasta 199,
- al 3º desde 200 hasta 299,
- ...
- al último le asignará los que queden hasta n.

El programa recogerá el resultado que calcule cada hilo, los irá sumando y, cuando todos hayan acabado, mostrará el resultado final.

```
La suma total de las raíces cuadradas desde 0 hasta 10875 es: 756105.62767
```

> **Recuperar el resultado**
>
> ¿Qué debe hacer el hilo principal para obtener el resultado de cada hilo?

Clase psp.actividades.U3A12_Calculador (hereda de Thread)

Esta clase recibirá como argumentos el número de hilo que es, el valor inferior del cálculo que debe realizar y el valor superior.

Con el número de hilo, se pondrá a sí misma un nombre (Calculador-X)

> **Comunicación entre hilos**
>
> ¿Cómo podemos hacer que el hilo devuelva la información al hilo principal?

Calculad el tiempo que tarda en realizar los cálculos variando la cantidad de hilos creados.

> **Tiempo de ejecución**
>
> Para calcular el tiempo que tarda en ejecutarse un proceso / hilo podemos usar el método System.currentTimeMillis() de la siguiente forma

```
long t_comienzo, t_fin;
t_comienzo = System.currentTimeMillis();
// Código que queremos monitorizar
t_fin = System.currentTimeMillis();
System.out.println("El proceso ha tardado " + (t_fin - t_comienzo) + "ms");
```

13. Clase psp.actividades.U3A13_Contador

Implementa un programa que reciba como argumentos una lista de ficheros de texto y que cuente el número de caracteres, palabras y líneas que hay en cada uno.

Para hacerlo, creará un hilo por cada archivo recibido que se encargará de procesar la información del archivo que reciba como parámetro.

Junto con la información obtenida, cada hilo mostrará el tiempo que ha tardado en completar la tarea.

14. Clase psp.actividades.U3A14_Cronometro

Crea una aplicación gráfica con Swing que simule un cronómetro.

En el diseño tendremos en la parte central de la ventana unas etiquetas representando hh:mm:ss

Tendremos un botón `Start / Resume` que comenzará / reanudará la cuenta del cronómetro. El valor mostrado se actualizará cada segundo.

Piensa que no podemos dejar al proceso principal bloqueado, por lo que *alguien* tendrá que encargarse de actualizar el valor cada segundo.

Tendremos un botón `Pause / Stop` que detendrá la cuenta del cronómetro pero mantendrá el valor actual. Si se vuelve a pulsar una segunda vez el botón reseteará el valor del cronómetro al valor inicial 00:00:00

Un flujo de ejemplo del programa podría ser este

> Cronómetro a 00:00:00 → Start / Resume → Empieza cuenta → Pause / Stop →
> Cronómetro a 00:00:37 → Start / Resume → Reanuda cuenta → Pause / Stop →
> Cronómetro a 00:01:05 → Pause / Stop → Cronómetro a 00:00:00