

Memoria de la práctica 1

Programación de Servicios y Procesos



24 de octubre de 2015

ÁLVARO GÓMEZ y daniel marcos

Profesor: José Manuel García

Índice

[Introducción 2](#_Toc433567911)

[**Antecedentes**: 2](#_Toc433567912)

[**Entorno de Desarrollo**: 2](#_Toc433567913)

[Enunciado Práctico 3](#_Toc433567914)

[**Explicación del Problema a resolver:** 3](#_Toc433567915)

[Solución 4](#_Toc433567916)

[**Código fuente de la Aplicación e Interfaces:** 4](#_Toc433567917)

[Explicación de: 6](#_Toc433567918)

[**Clases:** 6](#_Toc433567919)

[**Algoritmos:** 6](#_Toc433567920)

[**Estructuras de datos:** 6](#_Toc433567921)

[Listado de Pruebas 7](#_Toc433567922)

[**Entrada:** 7](#_Toc433567923)

[**Salida:** 7](#_Toc433567924)

[Conclusiones 8](#_Toc433567925)

Introducción

**Antecedentes**:

La siguiente memoria se enmarca dentro de una serie de guiones que se han realizado teniendo como objetivo la implementación del proyecto para la lectura del contenido de distintos usuarios, en el proyecto “conjunto”, para el Centro de Estudios Superiores Afuera, concretamente, para la Asignatura de Programación de Servicios y Procesos.

**Entorno de Desarrollo**:

NetBeans IDE 8.0.2

Enunciado Práctico

**Enunciado práctico:**

Supóngase que en el pc existe una carpeta por cada usuario: Usuario1 hasta Usuario5.

Crear un programa en java que cree varios procesos en paralelo.

Cada uno de ellos debe listar el contenido de la carpeta de usuario que recibe como argumento y escribir en el fichero de salida los archivos listados.

El final de la ejecución devuelve un fichero por cada usuario.

Una vez finalizados todos los hijos el padre imprimirá los ficheros por pantalla secuencialmente.

## **Explicación del Problema a resolver:**

Queremos desarrollar una aplicación en Java que, mediante la creación previa de carpetas para los distintos usuarios (con su información correspondiente), nos permita leer dichas carpetas para, posteriormente, mostrar todo su contenido.

Solución

## **Código fuente de la Aplicación e Interfaces:**

package conjunto;

import java.io.IOException;

import java.io.BufferedReader;

import java.io.FileReader;

import conjunto.Metodos;

/\*\*

\*

\* @author Daniel Marcos, Alvaro Gomez

\* @version 1.2 14/10/2015

\*/

public class main {

    // Variable global con el numero de usuarios

    static int nUsuarios = 0;

    /\*\*

    \* @param args the command line arguments

    \*/

    public static void main(String args[]) {

       // Llamos al metodo para ver los usuarios

       try {

           Metodos.leerUsuarios();

       } catch (Exception e) {

           e.printStackTrace();

       }

       // Llamamos al metodo para contar los usuarios

       try {

           nUsuarios = Metodos.contarUsuarios();

       } catch (Exception e) {

           e.printStackTrace();

       }

       // Creamos un array de procesos

       ProcessBuilder[] proceso = new ProcessBuilder[nUsuarios]; // Array de procesos

       // Bucle para crear los procesos (7), cada uno guardará en un documento el contenido de la carpeta de cada usuario

       try {

           BufferedReader leer = new BufferedReader(new FileReader("C:\\Users\\Alumnot\\Documents\\Usuarios\\usuarios.txt"));

           String user = null;

           int count = 0;

           while ((user = leer.readLine()) != null) {

               if (!(user.indexOf('.') >= 0)) {

                   Metodos.leerContenidoUsuarios(user, proceso[count]);

               }

               count++;

           }

           leer.close();

       } catch (Exception e) {

           e.printStackTrace();

       }

       // Bucle para imprimir por pantalla el fichero de cada usuario con toda la informacion que contenga

       try {

           Metodos.mostrarContenido();

       } catch (IOException e) {

           System.out.println("Error: " + e);

       }

    }

}

package conjunto;

import java.io.BufferedReader;

import java.io.File;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.IOException;

/\*

    \*@author Daniel Marcos, Alvaro Gomez

    \*@version 1.2 14/10/2015

\*/

public class Metodos {

    /\*\*

    \* Lanza un proceso hijo que recorre la ubicacion donde se encuentran los

    \* usuarios y almacena los nombres en un fichero de texto

    \*/

    public static void leerUsuarios() throws InterruptedException, IOException {

       // Declaramos el proceso con el comando a ejecutar

       ProcessBuilder pb = new ProcessBuilder("CMD", "/C", "dir /b C:\\Users\\Alumnot\\Documents\\Usuarios");

       pb.redirectOutput(new File("C:\\Users\\Alumnot\\Documents\\Usuarios\\usuarios.txt")); // Ubicacion de destino del resultado del comando

       //pb.redirectError(new File("C:\\Users\\Alumnot\\Documents\\Usuarios\\errorusuarios.txt")); // Ubicacion de destino si ocurre algun error

       Process p = pb.start(); // Lanza el proceso

       p.waitFor(); // Espera a que acabe

    }

/\*\*

    \* Metodo para contar los usuarios

    \*

    \* @return Numero de usuarios

    \*/

    public static int contarUsuarios() throws FileNotFoundException, IOException {

       // Abrimos el fichero creado anteriormente

       BufferedReader leer = new BufferedReader(new FileReader("C:\\Users\\Alumnot\\Documents\\Usuarios\\usuarios.txt"));

       // Variable donde se almacenara cada linea del fichero

       String linea = null;

       // Contador de usuarios

       int count = 0;

       // Si hay un registro de un archivo, que lleve un punto y una extension, la ignoramos

       while ((linea = leer.readLine()) != null) {

           if (!(linea.indexOf('.') >= 0)) {

               count++;

           }

       }

       leer.close(); // Cerramos el fichero

       return count;

    }

    /\*\*

    \* Lanza un proceso hijo que recorre la carpeta del usuario y almacena el

    \* indice del contenido en un fichero de texto

    \*

    \*/

    public static void leerContenidoUsuarios(String user, ProcessBuilder p) throws IOException, InterruptedException {

       p = new ProcessBuilder("CMD", "/C", "dir /b C:\\Users\\Alumnot\\Documents\\Usuarios\\" + user);

       p.redirectOutput(new File("C:\\Users\\Alumnot\\Documents\\Usuarios\\" + user + ".txt"));

       Process pr = p.start(); // Lanzamos el proceso

       pr.waitFor(); // Esperamos a que termine

    }

    public static void mostrarContenido() throws FileNotFoundException, IOException {

       BufferedReader leer = new BufferedReader(new FileReader("C:\\Users\\Alumnot\\Documents\\Usuarios\\usuarios.txt"));

       String user = null;

while ((user = leer.readLine()) != null) {

           if (!(user.indexOf('.') >= 0)) {

               BufferedReader carpetaUsuario = new BufferedReader(new FileReader("C:\\Users\\Alumnot\\Documents\\Usuarios\\" + user + ".txt"));

               String linea = null;

               System.out.println("El Usuario " + user + " contiene en su carpeta los siguientes archivos:");

               while ((linea = carpetaUsuario.readLine()) != null) {

                   System.out.println(linea);

               }

               carpetaUsuario.close();

           }

       }

       leer.close();

    }}

Explicación de:

## **Clases:**

El proyecto “conjunto” que hemos creado para la realización de la primera práctica de Programación de Servicios y Procesos, consta de dos clases:

1. La clase “main” que, como su propio nombre indica, contiene el núcleo principal del programa.
2. La clase “Metodos” que se encarga de albergar todos los métodos públicos que implementaremos, posteriormente, en la clase principal.

## **Algoritmos:**

leerUsuarios(): Lanza un proceso hijo que recorre la ubicación donde se encuentran los usuarios y almacena los nombres en un fichero de texto

contarUsuarios(): Método que utilizaremos para contar el número de usuarios, al finalizar, nos devolverá un contador, previamente declarado, con el número correspondiente.

leerContenidoUsuarios(): Lanza un proceso hijo que recorre la carpeta del usuario y almacena el índice del contenido en un fichero de texto.

mostrarContenido(): nos muestra el nombre de usuario con el correspondiente contenido que almacene dentro de su carpeta.

## **Estructuras de datos:**

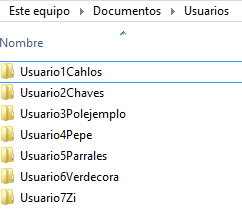
El sistema se basa en una organización de carpetas por usuario. Cada usuario tiene una carpeta y en ella se encuentran sus ficheros de texto.

Se crea un fichero de texto por usuario con el catálogo de su directorio.

Listado de Pruebas

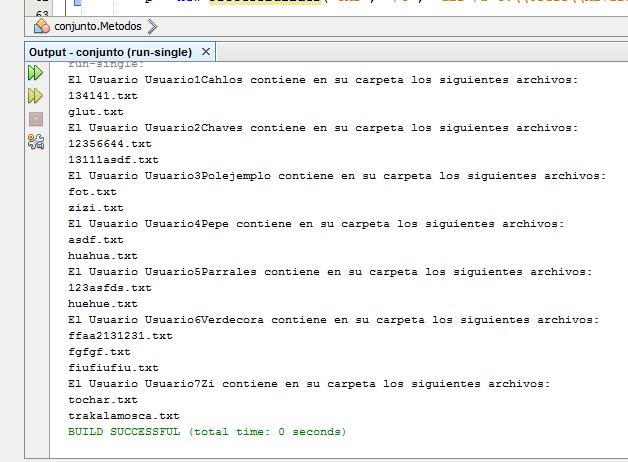
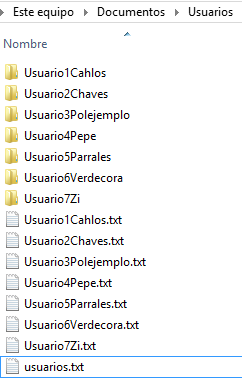
## **Entrada:**

Creación de carpetas para los usuarios en la ruta elegida

****

## **Salida:**

Se muestra por pantalla el resultado de la ejecución de la aplicación y la creación de los archivos de texto.

****

# Conclusiones

Para resolver la práctica hemos utilizado lo aprendido en los ejercicios prácticos del Tema 1, en especial la creación, comunicación y sincronización de procesos, y lectura y escritura en ficheros.

Hemos optado por añadir un paso más para obtener los nombres de las carpetas de usuario, para que se puedan asemejar más a usuarios reales teniendo nombres como Cahlos o Parrales, en vez de Usuario1 y Usuario2.

Al crear los ficheros en la misma carpeta donde están las carpetas de usuario nos surgió un imprevisto, porque al recorrerla nos imprimía también los ficheros, pero pudimos solventarlo filtrando los nombres, y si uno de ellos contenía una extensión y, por lo tanto un punto en el nombre, lo ignoraba.