**Programación de servicios y procesos**

**Tarea para PSP01.**

**Ejercicio 2**

* Primera parte: implementa una aplicación que escriba en un fichero indicado por el usuario conjuntos de letras generadas de forma aleatoria (sin sentido real). Escribiendo cada conjunto de letras en una línea distinta. El número de conjuntos de letras a generar por el proceso, también será dado por el usuario en el momento de su ejecución. Esta aplicación se llamará "lenguaje" y como ejemplo, podrá ser invocada así:

java -jar lenguaje 40 miFicheroDeLenguaje.txt

Indicando que se generarán 40 palabras del lenguaje y serán guardadas en miFicheroDeLenguaje.txt

* Segunda parte: implementa una aplicación, llamada 'colaborar', que lance al menos 10 instancias de la aplicación "lenguaje". Haciendo que todas ellas, colaboren en generar un gran fichero de palabras. Cada instancia generará un número creciente de palabras de 10, 20, 30, … Por supuesto, cada proceso seguirá escribiendo su palabra en una línea independiente de las otras. Es decir, si lanzamos 10 instancias de "lenguaje", al final, debemos tener en el fichero 10 + 20 + 30 + … + 100 = 550 líneas.

Para la primera parte he intentado aprovechar los ejemplos de la plataforma. Como en el primero, he intentado comentarlo y documentarlo lo mejor posible para una mejor comprensión y asimilar conceptos de cara a futuros usos.

/\*

\* TAREA PARA PSP01 EJERCICIO 2.

\* PRIMERA PARTE:

\* Implementa una aplicación llamada 'lenguaje', que escriba en un fichero

\* indicado por el usuario conjuntos de letras generadas de forma aleatoria

\* (sin sentido real).

\* Escribiendo cada conjunto de letras en una línea distinta y con de un número

\* de letras dado por el usuario en el momento de la ejecución.

\*/

package lenguaje**;**

**import** java**.**io**.**File**;**

**import** java**.**io**.**IOException**;**

**import** java**.**io**.**RandomAccessFile**;**

**import** java**.**nio**.**channels**.**FileLock**;**

**import** java**.**util**.**Random**;**

/\*\*

\*

\* **@author** juang <juangmuelas@gmail.com>

\* **@since** 02/11/2020

\*/

public class Lenguaje **{**

/\*\*

\* **@param** args the command line arguments

\*/

public static void main**(**String**[]** args**)** **{**

/\*\*

\* Siguiendo el consejo del tutor, se reaprovecha parte de los ejemplos

\* de accesos sincro, usando objeto RandomAccessFile

\*/

/\*\*

\* **@param** args numPalabras argumento a pasar para crear archivo

\* **@param** args nombreFichero argumento a pasar para crear archivo

\*/

int numPalabras**=**0**;**

String nombreFichero**=**""**;**

/\*\*

\* **@param** args fichero variable para la creación del objeto File

\*/

File fichero **=** **null;**

/\*\*

\* **@param** args raf variable dea acceso para el objeto RandomAcessFile

\*/

RandomAccessFile raf **=** **null;**

/\*\*

\* **@param** args bloqueo variable para el control de acceso

\*/

FileLock bloqueo **=** **null;**

/\*\*

\* El ejercicio pide que la aplicacion solicite número de palabras y

\* fichero. Siguiendo ejemplo dado, damos al condicional la necesidad

\* de pasar los dos argumentos y dentro, evitamos consultar el SO

\*(fuera a priori de las indicaciones del ejercicio propuesto).

\*/

**if** **(**args**.**length**==**2**){**

**try{**

numPalabras **=** Integer**.**parseInt**(**args**[**0**]);**

nombreFichero **=** args**[**1**];**

// Creamos un objeto File para acceder al fichero

//Antes por ordenar el código declaramos la variable al inicio

fichero **=** **new** File**(**nombreFichero**);**

/\*\*

\* Tras evitar las consultas del SO, la siguiente nos lleva

\* a acceder al fichero, pero antes se debe verificar si este

\* existe o debe crearse.

\*/

**if** **(!**fichero**.**exists**()){**

//Si no existe el fichero

**try** **{**

fichero**.**createNewFile**();** //Lo creamos

**}catch(**Exception e**){**

System**.**err**.**println**(**"ERROR EN LA CREACIÓN DEL FICHERO"**);**

System**.**exit**(**1**);** //Si hay error, finalizamos

**}**

**}**

**try{**

raf **=** **new** RandomAccessFile**(**fichero**,**"rw"**);** //Abrimos el fichero

//\*\*\*\*\*\*\*\*\*\*\*\*\*

//Sección crítica

bloqueo **=** raf**.**getChannel**().**lock**();**

//bloqueamos el canal de acceso al fichero

**for** **(**int a**=**0**;** a**<**numPalabras**;** a**++){**

Random r **=** **new** Random**();**

//Añadimos por control una limitación de letras por palabra

int nLetras**=** r**.**nextInt**(**20**)+**1**;**

//Ahora el número aleatorio de letras por palabra

**for** **(**int i**=**0**;** i**<**nLetras**;** i**++){**

//nos colocamos al final del fichero para añadir valores

raf**.**seek**(**raf**.**length**());**

String**[]** abc **=** **{**

"A"**,**"B"**,**"C"**,**"D"**,**"E"**,**"F"**,**"G"**,**"H"**,**"I"**,**"J"**,**"K"**,**

"L"**,**"M"**,**"N"**,**"Ñ"**,**"O"**,**"P"**,**"Q"**,**"R"**,**"S"**,**"T"**,**"U"**,**

"V"**,**"W"**,**"X"**,**"Y"**,**"Z"**};**

int numLetra **=** r**.**nextInt**(**26**);**

//escribimos la letra conseguida en el fichero

raf**.**writeBytes**(**abc**[**numLetra**]);**

**}**

//El ejercicio pide palabras en linea distinta.

raf**.**seek**(**raf**.**length**());**

raf**.**writeChar**(**'\n'**);**

**}**

System**.**out**.**println**(**"Proceso terminado"**);**

bloqueo**.**release**();** //liberamos el bloqueo del fichero

bloqueo **=** **null;**

//Fin sección crítica

//\*\*\*\*\*\*\*\*\*\*\*\*\*

**}catch(**IOException eio**){**

System**.**err**.**println**(**"Erroral acceder al fichero"**);**

System**.**err**.**println**(**eio**.**toString**());**

**}**

**}catch(**NumberFormatException ex**){**//Errores de formato

System**.**out**.**println**(**"Debe indicar el número de palabras y nombre de fichero"**);**

System**.**err**.**println**(**ex**.**toString**());**

**}**

// CERRAR FICHERO.

**try{**

**if** **(null** **!=** raf**)** raf**.**close**();**

**if** **(null** **!=** bloqueo**)** bloqueo**.**release**();**

**}** **catch** **(**Exception e2**)** **{**

System**.**err**.**println**(**"Error al cerrar el fichero"**);**

System**.**err**.**println**(**e2**.**toString**());**

System**.**exit**(**1**);** //Si hay error, finalizamos

**}**

**}else{**

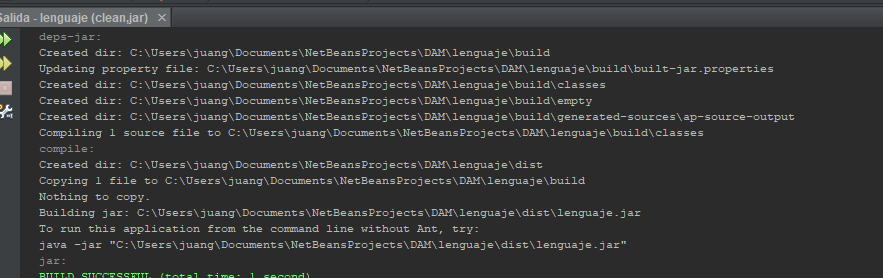
System**.**out**.**println**(**"Debe indicar el número de palabras y nombre de fichero"**);**

**}**

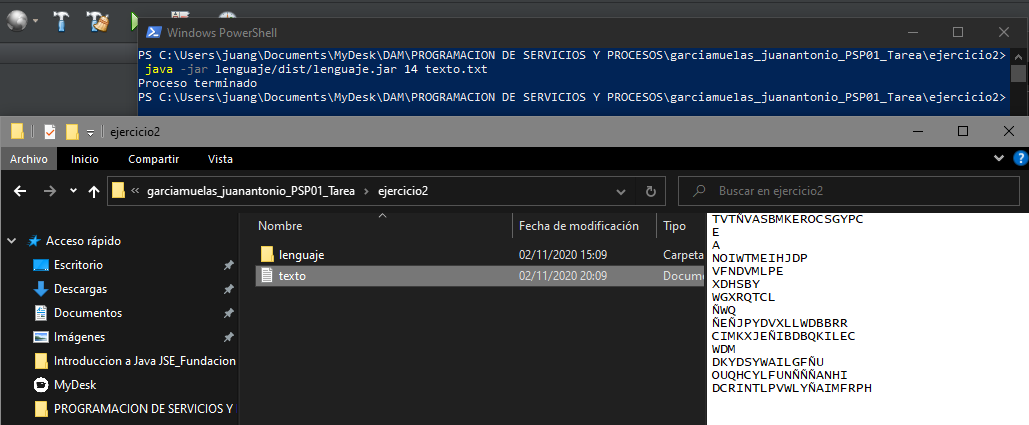
**}** //fin clase Main

**}**//Fin clase lenguaje

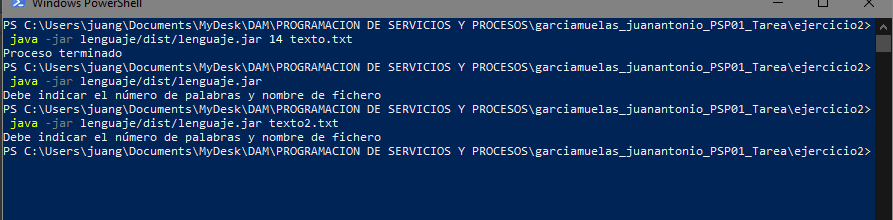
Creamos el archivo .jar



Probamos la ejecución en la consola de Windows.



Y los posibles errores.



Para la segunda parte aprovecho una gran cantidad del código de la plataforma del punto 6.1 accesos múltiples.

También lo más comentado y documentado posible.

/\*

\* TAREA PARA PSP01 EJERCICIO 2.

\* SEGUNDA PARTE:

\* Implementa una aplicación llamada 'colaborar', que lance al menos 10

\* instancias de la aplicación "lenguaje", formando un gran fichero.

\* Cada instancia generará un número creciente de palabras de 10,20,30...

\* Escribiendo cada conjunto de letras en una línea distinta y con de un número

\* de letras dado por el usuario en el momento de la ejecución.

\*/

package colaborar**;**

**import** java**.**util**.**Random**;**

/\*\*

\*

\* **@author** juang <juangmuelas@gmail.com>

\* **@since** 02/11/2020

\*/

public class Colaborar **{**

/\*\*

\* **@param** args the command line arguments

\*/

public static void main**(**String**[]** args**)** **{**

/\*\*

\* **@param** args totalInstancia integer que guarda el número a añadir

\*/

int totalInstancias **=** 0**;**

/\*\*

\* **@param** args palabras integer que guarda el número de palabras por

\* instancia. Inicializamos en 10 para seguir el ejercicio propuesto

\*/

int palabras **=** 10**;**

/\*\*

\* **@param** args nuevoProceso variable de tipo Process

\*/

Process nuevoProceso**;**

//Procedemos a añadir instancias en bloques de 10

**while** **(**totalInstancias**<**10**){**

Random r **=** **new** Random**();**

//Añadimos una limitación para no extender la prueba

totalInstancias **=** r**.**nextInt**(**50**);**

**}**

//Comprobamos que solo recibe el total a ejecutar

**if(**args**.**length**==**1**){**

/\*\*

\* seguimos el código del ejemplo múltiples accesos, pero al

\* tener ya la primera parte hecha, evitamos el paso de asegurar

\* la existencia del fichero y creamos las instancias que accederán

\* al mismo.

\*/

**try{**

**for** **(**int i **=** 1**;** i **<=** totalInstancias**;** i**++){**

nuevoProceso **=** Runtime**.**getRuntime**().**exec**(**"java -jar "**+**

"lenguaje.jar " **+** palabras **+** " "**+** args**[**0**]);**

//Creamos el nuevo proceso y le indicamos el número de palabras y

//el fichero que debe utilizar.

System**.**out**.**println**(**"Creada la instancia " **+** i**);**

//Mostramos en consola que hemos creado otro proceso

palabras **=** palabras **+** 10**;** //que aumentarán de 10 en 10

**}**

**}catch** **(**SecurityException ex**){**

System**.**err**.**println**(**"Ha ocurrido un error de Seguridad."**+**

"No se ha podido crear el proceso por falta de permisos."**);**

**}catch** **(**Exception ex**){**

System**.**err**.**println**(**"Ha ocurrido un error, descripción: "**+**

ex**.**toString**());**

**}**

**}else{**

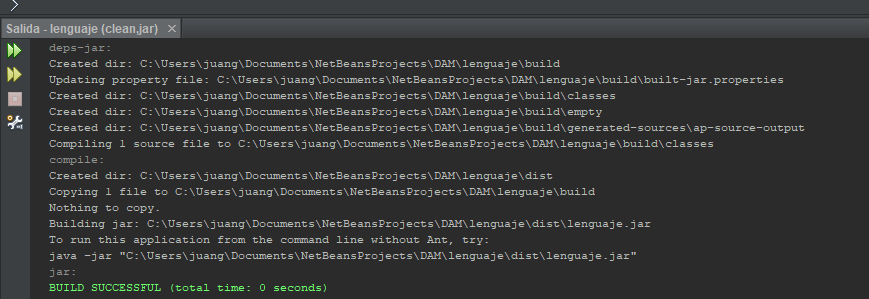
System**.**out**.**println**(**"Indique la ruta del archivo"**);**

**}**

**}**//Fin de main

**}**//Fin clase Colaborar

Luego, creamos el archivo .jar



Probamos la ejecución, probando error en la llamada

