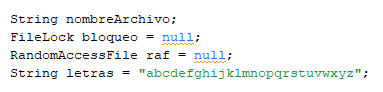
Para el ejercicio 2 he creado dos proyectos a los que he llamado Ejercicio2\_p1 y Ejercicio2\_p2

En el proyecto Ejercicio2\_p1 he creado una clase main llamada *Lenguaje* dónde he diseñado el código necesario para que la aplicación genere un fichero con una cantidad de cadenas de caracteres aleatorios igual a la cantidad especificada al ejecutar el programa.

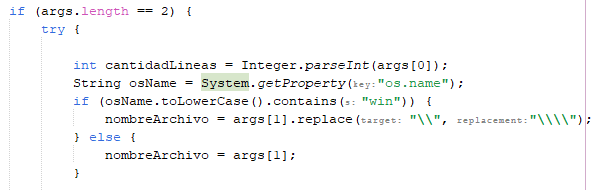
Para ello en primer lugar he creado una variable de tipo String llamada *nombreArchivo* donde se guardará el segundo argumento (el nombre del archivo que se generará) que recibirá el programa. También he creado un objeto de la clase FileLock llamado *bloqueo* que usaremos para bloquear el canal de acceso al fichero y luego liberarlo y un objeto de la clase RandomAccessFile llamado *raf* para acceder al fichero con los permisos que le especificaremos. También he creado un String llamado *letras* donde escribiré el alfabeto. (Para crear esos objetos he tenido que importar las clases pertinentes).



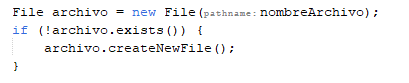
Tras declarar estos objetos, he creado un If donde indico que, si el programa recibe dos argumentos, continue con normalidad, pero si no recibe dos entonces mostrará un mensaje informando de que debe recibir dos argumentos.

Si recibe dos argumentos entonces el programa crea una variable llamada *cantidadLineas* de tipo entero y la inicializa con un valor igual al primer parámetro recibido, que indica la cantidad de líneas que debe generar el programa.

Como la ruta del archivo varía en función del sistema operativo utilizado, he creado una variable llamada *osName* de tipo String donde guardaremos el nombre del sistema operativo usado en el momento de ejecución. Y a continuación he creado un If para que, si el sistema operativo es Windows, guarde el path en la variable *nombreArchivo* y reemplace las barras de la ruta para evitar errores; y, si no es Windows, la guarde sin reemplazarlas.

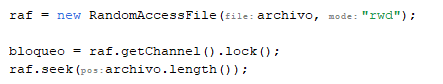


A continuación, he creado un objeto de tipo File llamado *archivo* al que le pasamos como parámetro el path que hemos guardado en *nombreArchivo* y le indicamos que, si el archivo no existe, lo cree.

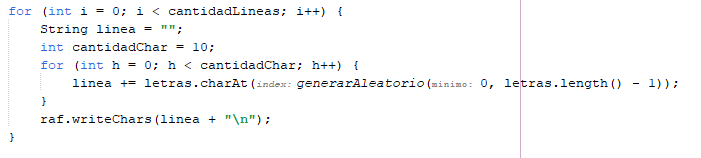


Iniciamos la variable raf (RandomAccessFile) creando el objeto dándole como parámetros el objeto File que hemos creado y le indicamos los permisos “rwd”.

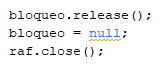
Iniciamos también el bloqueo para proteger el recurso. Y le indicamos también dónde nos situamos en ese fichero, que en este caso será al final.



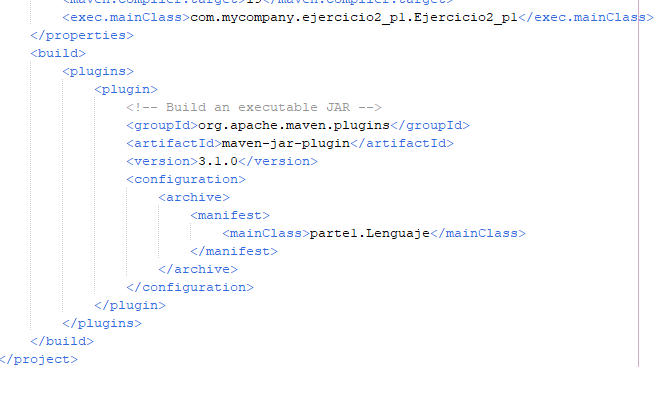
Luego he creado un bucle For que se repetirá una cantidad de veces igual a la variable *cantidadLineas* que indicamos antes y en cada vuelta el buble creará un String llamado *linea*. Como el ejercicio no indica cuantos caracteres deben tener las palabras generadas, he puesto que tenga 10 creando una variable llamada *cantidadChar*. Dentro de este bucle, creamos otro que se repetirá una cantidad de veces igual al numero de letras que tendrá la cadena y le indico que, en cada vuelta, añada a *línea* un carácter que sacará del String *letras* y para ellos he usado el mismo método que utilicé en el ejercicio anterior para conseguir un número aleatorio. Al terminar este segundo bucle, hará un salto de línea para que cada cadena de caracteres generada se encuentre en una línea distinta.



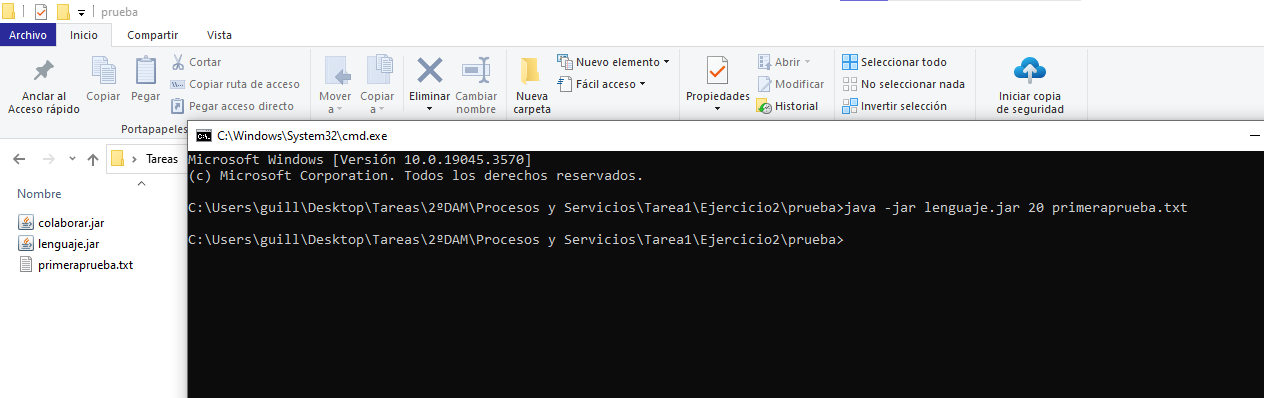
Para terminar, liberamos el bloqueo y cerramos el objeto *raf*

**

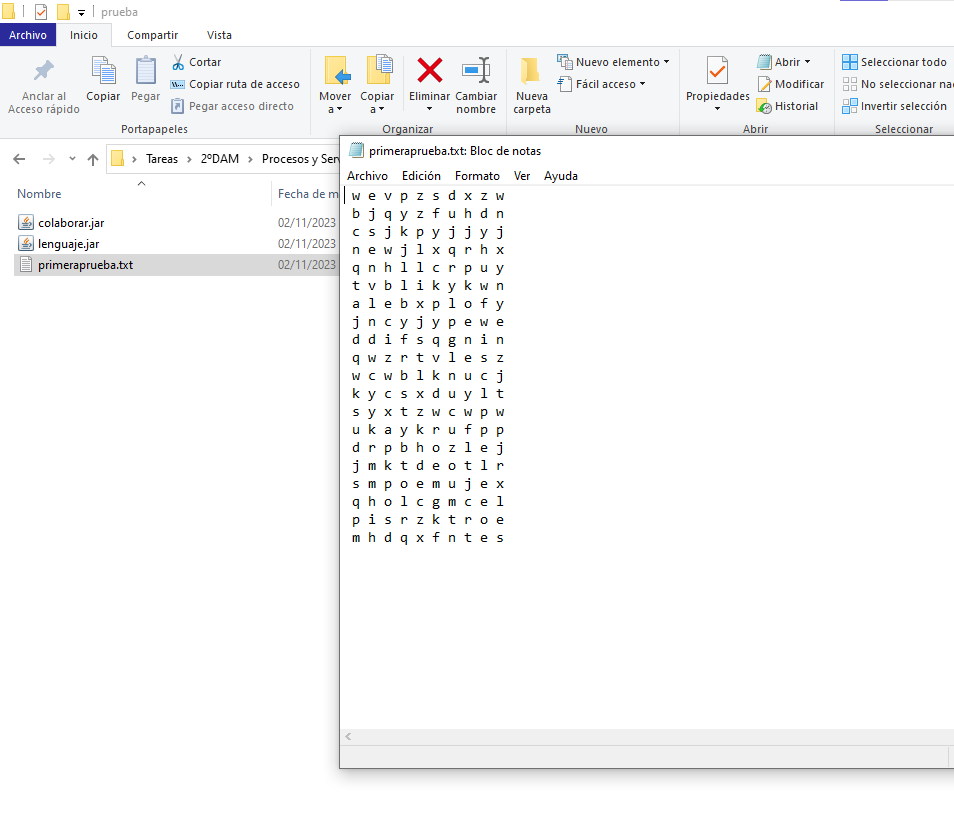
Ahora nos dirigimos al archivo pom.xml del proyecto y añadimos el plugin que le permite ejecutarse con el .jar. Especificamos cual es la clase principal que debe ejecutarse.



Para probarlo, he creado una carpeta llamada prueba donde he metido el archivo .jar que hemos generador al pulsar Clean and Build. Abro la consola en esa ubicación y escribo el comando para ejecutar el programa, añadiendo los dos argumentos (la cantidad de líneas que quiero y el archivo que debe generar) *java -jar lenguaje.jar* 20 primeraprueba.txt



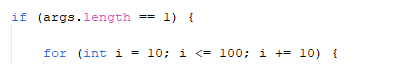
Vemos que nos ha creado el fichero en esa misma ubicación. Lo abrimos para comprobar que ha funcionado



Ahora, para crear el segundo programa, en el proyecto Ejercicio2\_p2 he creado una clase main llamada *Colaborar* dónde he diseñado el código necesario para que la aplicación pida que se generen 10 instancias del primer programa para que terminemos consiguiendo un archivo de texto con 550 lineas.

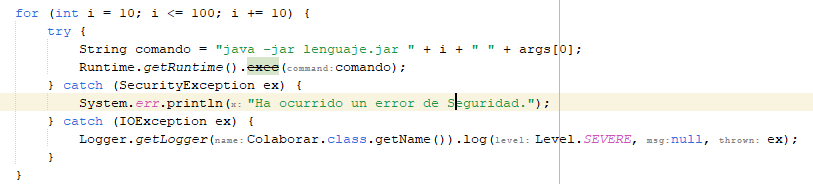
Para ello, en primer lugar he creado un If donde indico que si el programa recibe un argumento (el path del fichero a generar), continue con normalidad, pero si no recibe uno entonces mostrará un mensaje informando de que debe recibir un argumento.

A continuación creo un buble For en el que declaro que la variable *i*  = 10 y que se repetirá mientras que *i* sea igual o menor que 100, y a cada vuelta *i* se incrementará en 10. De esta forma conseguimos que la variable *i* (que luego usaremos para indicar el primer argumento del programa *Lenguaje*) tenga un valor de 10, luego de 20, 30, 40… hasta 100.



Dentro de ese bucle, he creado un String llamado *comando* que replica el comando que usé en la consola para probar la primera aplicación, donde *i* será el primer argumento (la cantidad de líneas) y el argumento recibido en esta segunda aplicación (el path del fichero) será el segundo argumento para la primera aplicación.

Tras esto, con la sentencia Runtime.getRuntime().exec(comando) le indicamos que ejecute el comando guardado en el String. Todo esto incluido dentro de un try/catch que he incluido un SecurityException para los casos donde se produzca un error de seguridad por no tener los permisos necesarios para acceder al recurso.

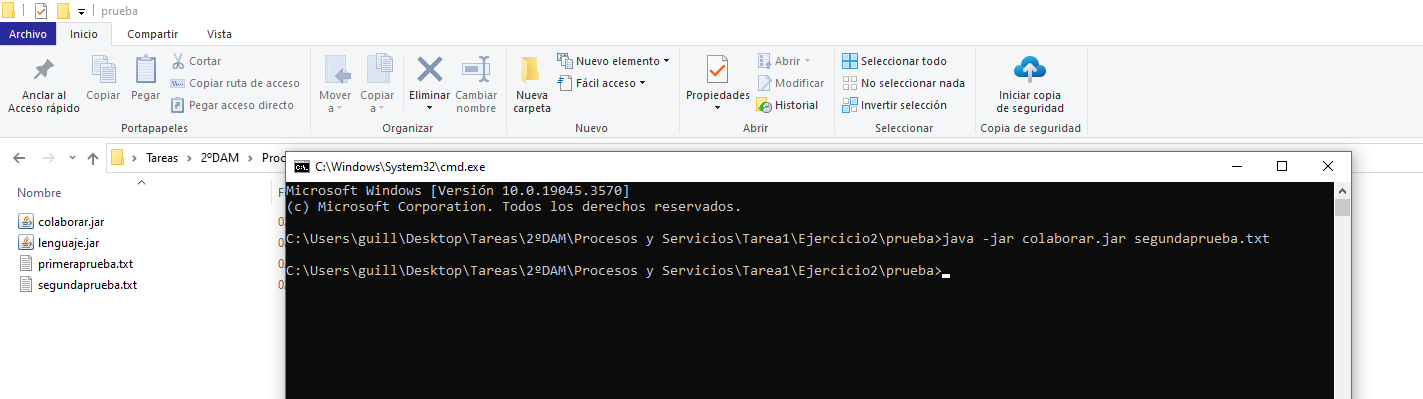


Para terminar, al igual que hicimos con le primer programa, modificamos el pom.xml para incluir el plugin que permite ejecutar el archivo .jar.



Para probarlo, copio el archivo .jar que hemos generador al pulsar Clean and Build en la carpeta *prueba* donde teníamos el .jar de la primera aplicación.

Abro la consola en esa ubicación y ejecuto el comando *java -jar colaborar.jar segundaprueba.txt*



Abrimos el archivo y comprobamos que nos ha generado un documento con 550 líneas de cadenas de letras aleatorias.

