ESCUELA POLITÉCNICA SUPERIOR DE CÓRDOBA

Entregable Práctica 1

Práctica 1 – Iniciación a Java

Trabajo realizado por:

Pedro Pablo García Pozo
Rubén Borrego Canovaca

Decisiones y dificultades de diseño

En este apartado vamos a explicar de forma breve las decisiones que hemos tomado a lo largo de la realización del proyecto para ir solucionando las diferentes dificultadas con las que nos íbamos encontrando.

Empezamos con el gestor de contactos. Para ello, decidimos primeramente crear la clase contacto con todos su atributos y métodos para así poder tener un punto de partida desde donde comenzar la implementación puesto que como nunca habíamos programado con java, a la hora de diseñar el programa en cuestión no sabíamos muy bien si todos los métodos que usábamos en el lenguaje c estarían en java por lo cual era mejor programar antes de diseñar mucho del programa. La primera dificultad fue el patrón de diseño Singleton que debíamos utilizar en el gestor de contactos, con el cual nunca habíamos trabajado pero que fue bastante fácil de implementar gracias a los apuntes que nos fueron dados. Una vez ya teníamos el tema de Singleton zanjado, el siguiente obstáculo con el que nos encontramos fue el tema de los intereses, los cuales deberían de ser únicos e irrepetibles, para lo cual pensamos primero en usar Enums, pero tuvimos que desechar la idea ya que en la práctica se imponía que los intereses deberían de leerse del fichero .properties y con Enums no se puede ya que deberían de estar declarados en tiempo de compilación, por lo cual decidimos usar un ArrayList de Strings para ello creando todos los métodos necesarios para que no hubiera intereses repetidos y para que funcionase todo bien. Una vez que el programa funcionaba bien por consola, decidimos pasar a la implementación de las funciones para guardar en ficheros planos, ya que es lo más fácil para novatos en java como nosotros. Para ello decidimos crear la función toStringFile para dar un formato correcto a la hora de escribir y leer del fichero. Para poder leer cada variable por separado, pero cogiendo una línea cada vez, investigamos y dimos con la clase StringTokenizer que permitía dividir en diferentes Strings uno solo usando para ello un separador previamente estipulado por nosotros y que decidimos que fuera el "|" (aprovechamos para decir que cuando haya que escribir algo por teclado no uséis el carácter "|" puesto que si no el fichero daría problemas). La lectura y guardado de los

datos se produce al principio y al final de la ejecución y tras realizar las pruebas pertinentes quedo todo operativo.

Tras esto proseguimos con el gestor de anuncios. Para ello obviamente comenzamos con la creación de la clase anuncio(post) y de dos Enums para el tipo de Anuncio. En los requisitos se nos especificó que debíamos de usar el patrón Factory el cual fue un gran obstáculo al principio ya que debíamos de establecer ciertos atributos general en el Supertipo y luego crear subtipos con los atributos específicos necesarios para cada caso (individualizado, temático y flash). Para ello primero creamos las clases que extendían la clase Post y en las cuales también agregamos los ToString y los ToStringFile para ir preparando la posterior escritura en otro fichero de los anuncios. Tras leernos bien los ejemplos del profesor y tras preguntarle en clase damos con la solución para el problema que teníamos con la Factory que era en que forma íbamos a pasarle los argumentos a los constructores, y que solucionamos creando un constructor genérico en la Factory con todas las posibles variables, para así pasarle los distintos argumentos necesarios para cada subtipo en los correspondientes huecos y en los demás que no había que utilizar pasándole Null. Tras esto y tras implementar todos los métodos necesarios en el gestor de anuncios el siguiente tropezón lo tuvimos en el tema de las fechas publicación ya que no se asignarían a no ser que el anuncio se publicara y por lo tanto a la hora de leer y escribir de ficheros (para la cual prácticamente copiamos la implementación que hicimos en el gestor de contactos) había ocasiones en que en la fecha de publicación había un Null y con el tema de la clase Date no hacía bien el Parse y el programa daba error. Tras solventar esto el ultimo problema lo tuvimos con la ordenación del tablero; de los dos tipos de ordenación la de por usuario propietario era la más fácil, pero el problema llego en la otra que era por fechas. Tras volver a pegarnos un buen rato investigando en internet dimos con la solución que era la función collections.sort con la cual se nos ordenaba un Array auxiliar que creamos con los anuncios que el programa tenía que enseñar en el tablón. Tras comprobar que todo funcionaba solucionamos algunos errores menores y terminamos añadiendo en la función de lectura del fichero unas líneas de código para manejar el tema de los anuncios flash, ya que si al abrir el fichero un anuncio está en espera y la fecha actual está en el rango tendría que postearlo de forma automática, así como a su vez si el estado del anuncio es publicado y la fecha ya ha pasado también tendría que archivarlo de forma

automática lo cual resulto en un completo éxito. Tras hacer todo esto compilamos y generamos el ejecutable usando el NX para entrar de forma remota a los servidores de la UCO. Después de esto generamos el Javadoc el cual fuimos haciendo de forma progresiva mientras íbamos codificando el programa y que también logramos sin mayores problemas.

Restricciones de uso

En este apartado voy a explicar de forma breve algunas situaciones que pueden degenerar en problemas en nuestro programa.

La primera y más importante es la de modificar ficheros de forma manual. Los ficheros ya vienen incluidos en el zip y están inicialmente vacíos. Para su correcto funcionamiento es mejor no escribir nada ni tocarlos ya que se pueden corromper al introducir algún espacio en algún lugar erróneo. En caso de querer borrar todos los datos o de que por algún casual al tocar el fichero da algún error, se puede perfectamente borrar cualquiera de los ficheros ya que estos se volverán a crear cuando detecte que no existen.

La segunda es cuando por ejemplo en el caso de los intereses o de los destinatarios el programa pide una o más variables, se deberán introducir por teclado separadas por comas, en el caso de usar espacios entre comas no hay ningún problema ya que tratamos lo escrito por teclado para eliminar espacios.

La tercera es que, en el gestor de anuncios, los usuarios se identifican con el correo ya que el nombre no es una buena clave.

Por último, a la hora de introducir fechas o de elegir opciones en los menús habrá que seguir los formatos previamente estipulados para que el programa no tienda a errores, como por ejemplo no usar letras en vez de números cuando haya que elegir una opción y no introducir fechas incompletas o con otro formato por teclado, aunque en el caso de las fechas el tratamiento de errores funciona de forma excelente.

Conclusión

Como conclusión hemos de decir que hemos aprendido bastante de los patrones de diseño y sobre todo de java, y que, aunque hemos tenido algunas dificultades bastante difíciles de solventar a priori, hemos podido solucionarlo y hacer que el programa funcione de forma correcta sin más problemas.

Bibliografía/Fuentes

https://docs.oracle.com/
https://www.w3schools.com/
https://stackoverflow.com/
https://stackoverrun.com/