

DISEÑO DE INTERFACES DE USUARIO CURSO 2019/20

PRÁCTICA 8 – COMPRIMIR ARCHIVOS EN FORMATO ZIP

Miembros del grupo:

Nassr Eddine Moussati Lamhamdi Yousuf Boutahar El Maachi

Índice

[Introducción 3](#_Toc24399676)

[Desarrollo 4](#_Toc24399677)

[Autoevaluación de acuerdo a los principios de Schneiderman y Plaisant. 6](#_Toc24399678)

[Conclusiones 9](#_Toc24399679)

# Introducción

En la práctica se nos ha propuesto desarrollar una interfaz cuya funcionalidad es la de realizar una compresión de un directorio seleccionado por el usuario y volcar esos archivos comprimidos en otra dirección destino elegida también por el usuario en la interfaz.

Para generar el archivo comprimido .zip utilizaremos el paquete de clases java.util.zip que proporciona java.

La funcionalidad que deberá tener la aplicación a desarrollar en esta práctica será la siguiente:

* Permitir al usuario seleccionar una **carpeta existente** (uso de la clase

*JFileChosser*), la cual será la carpeta que contiene todos los archivos a

comprimir.

* Permitir al usuario seleccionar una **carpeta existente de salida** (uso de la

clase *JFileChosser*), la cual será la carpeta donde se genera el archivo

**folder.zip**.

* Una vez seleccionadas la carpetas, y comprobada que nos son la misma

carpeta (por razones obvias), la aplicación debe permitir mediante un botón

(*JButton*) **iniciar la tarea de comprimir los archivos en segundo plano** (uso

de la clase *SwingWorker*).

* Se debe mostrar el progreso de la tarea que está comprimiendo los archivos

mediante una Barra de Progreso (uso de la clase *JProgressBar*).

* La aplicación debe permitir al usuario cancelar la generación del archivo zip

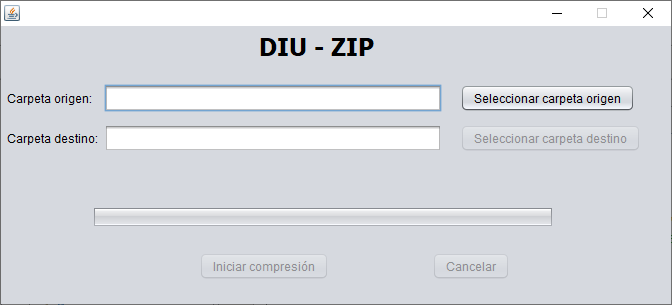
mediante un botón de Cancelar (JButton), el cual sólo estará habilitado cuando

se esté generando el archivo comprimido zip.

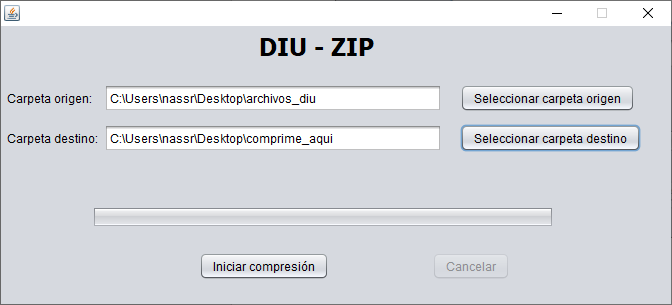
# Desarrollo

En la práctica se nos pidió desarrollar una interfaz en la que con el uso de de los swing workers, donde se mostraba como se ejecutaban tareas en segundo plano para que la aplicación principal no se quedase bloqueada por la ejecución de X tareas. Además se debía de mostrar el transcurso de la compresión con un JProgressBar.

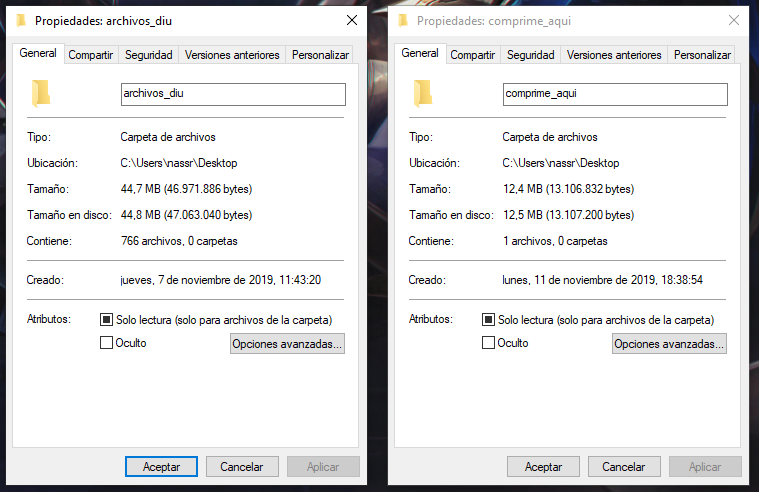
Vista principal:



Vista con los directorios correspondientes a comprimir:



Antes de la compresión y después de la compresión:



El flujo principal del programa es el siguiente, el usuario selecciona la carpeta que se quiere comprimir mediante el JButton de selección de ruta de origen y esta es mostrada al lado en el JTextField, luego el usuario realiza la misma acción, pero esta vez elige la ruta de destino a la que ira el archivo .zip

Luego el usuario procederá a la realización del proceso de compresión desde el JButton pertinente, si no ocurren excepciones como por ejemplo, que las rutas origen y destino sean la misma carpeta, la interfaz realizara dicho proceso sin problemas y el usuario podrá ver el resultado en su sistema de fichero local. Decidimos también que si el usuario consideraba cancelar la compresión del archivo, el archivo comprimido generado hasta el momento fuese eliminado ya que vimos que el comportamiento era parecido en otros compresores zip.

Cabe destacar que durante el desarrollo tuvimos un problema con la JProgressBar y es que no conseguimos borrar el transcurso mostrado si el usuario decidía cancelar la compresión.

# Autoevaluación de acuerdo a los principios de Schneiderman y Plaisant.

1. Consistencia:

En la aplicación hemos desarrollado una interfaz de acuerdo a la disposición de los diferentes componentes;

* En la parte superior encontramos 2 JTexFields para que el usuario pueda elegir los directorios de origen y destino deseados para la compresión.
* Una barra horizontal JProgressBar que muestra el progreso de manera visual al usuario.
* En la parte inferior de la interfaz dos botones de realizar la compresión y la de cancelar.

Dichos componentes se han posicionado de acuerdo a como un usuario corriente le resultaría más familiar y amigable el uso de ésta, esto es posible de lograr si sabemos con qué tipo de interfaces está familiarizado el usuario.

1. Usabilidad Universal:

Creemos que debido a la sencillez de la interfaz desarrollada es fácilmente interpretable por cualquier persona independientemente de su edad o procedencia, es una interfaz intuitiva que todos sabrían como interactuar con ella.

1. Ser Informativo:

Es importante informar sobre el uso de una interfaz, pero tampoco hay que excederse demasiado, en nuestra interfaz hemos creído necesario la señalización de los JTextFields que muestran las rutas origen y destino para facilitar al usuario el funcionamiento de dichos campos.

1. Flujo de tareas:

En nuestra implementación hemos desarrollado la interfaz de manera que el usuario se sienta visualmente cómodo y respondido por parte de la interfaz al realizar cualquier cambio en ella, es decir, cuando el usuario por ejemplo selecciona la ruta en el sistema de ficheros emergente, se muestra automáticamente en los campos de texto señalizados como rutas de origen y destino.

1. Prevenir errores:

Se han tomado medidas para que el usuario no pueda realizar acciones prohibidas como por ejemplo; los campos de texto para las rutas no son editables para que el usuario no pueda insertar nada mas que directorios validos mediante el JFileChooser que muestra el sistema de ficheros local del usuario.

Por otro lado se han controlado todos los errores posibles que puedan ocurrir en la interfaz por parte del usuario como por ejemplo, que el usuario intente realizar la compresión sin tener primero las rutas seleccionadas, o que el usuario no pueda elegir la misma carpeta de origen y de destino para realizar la compresión.

1. Facilitar deshacer:

Hemos implementado la aplicación de manera modular separando las funcionalidades de la interfaz de manera que ante cualquier intención de cambiar la funcionalidad o de eliminar alguna, no haya ningún problema.

Por otro lado, se ha desarrollado la funcionalidad de detener el proceso de compresión mediante un JButton en la interfaz por parte del usuario

1. El usuario debe sentir el control:

El usuario al ejecutar la aplicación e interactuar con la interfaz siente la facilidad de manejar las funcionalidades que se le brindan al completo con el simple hecho de pulsar algún botón, la interfaz responde de manera directa haciendo que el usuario sienta control sobre ésta.

1. Sin memoria a corto plazo:

La interfaz carece de mucha información para que el usuario no se sienta ofuscado ni con la necesidad de buscar cómo funciona al usar la interfaz.

Además, si por algún casual el usuario realiza alguna operación mal, la interfaz le facilitara la información necesaria mediante mensajes emergentes para poder hacer uso de manera correcta la interfaz.

# Conclusiones

Consideramos que la práctica lo que intenta transmitir desde el punto de vista del usuario, es la relación que existe entre distintos componentes de la interfaz y que sea intuitivo a simple vista la funcionalidad de la aplicación.

Por otra parte, también tuvimos en cuenta que intentar innovar en exceso en la interfaz no es muy productivo ya que el usuario final acaba teniendo referencia de otros programas y si la interfaz no es muy clara, ésta sería inútil.