# PROGRAMACIÓN ORIENTADA A OBJETOS

**I/O 201902**

**Laboratorio 6/6 [ :) ]**

## DESARROLLO

**Preparando**

En este laboratorio vamos a extender el proyecto teatroColon adicionando un menú barra con las opciones básicas de entrada-salida y las opciones estándar de iniciar y salir.

1. En su directorio descarguen la versión del proyecto realizado por ustedes para el

laboratorio 03 y preparen el ambiente para trabajar desde **CONSOLA**

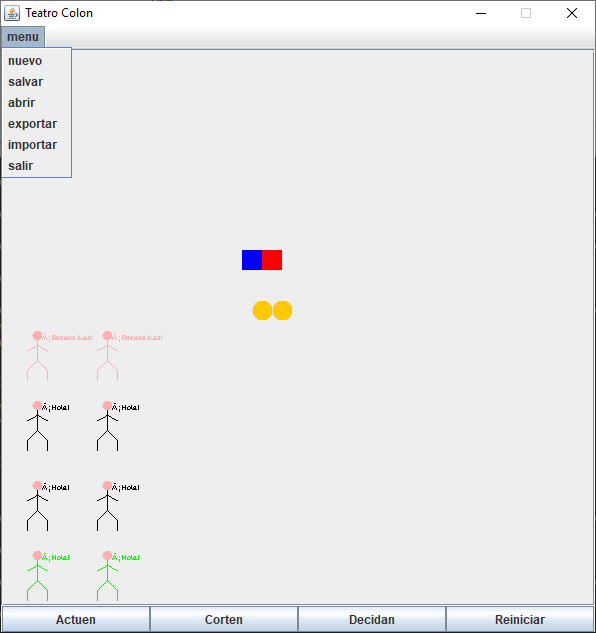
1. Ejecuten el programa, revisen la funcionalidad.

## Creando la maqueta

**[En lab06.doc, \*.asta y \*.java] [NO OLVIDEN BDD y MDD]**

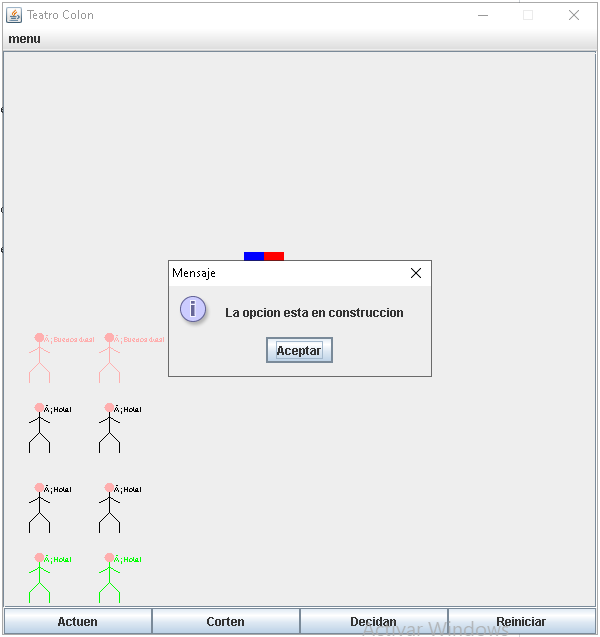
En este punto vamos a construir la maqueta correspondiente a esta extensión siguiendo el patrón MVC.

1. **MODELO:** Preparen los métodos correspondientes a reiniciar y a las cuatro opciones básicas de entrada-salida (salve,abra,exporte,importe). Los métodos deben simplemente propagar una teatroColonExcepcion con el mensaje: “Opción … en construcción”. Los métodos de entrada salida deben tener un parámetro File.
2. **VISTA :** Construyan un menú barra que ofrezca, además de las opciones básicas de entrada-salida, las opciones estándar de iniciar y salir. Para esto creen el método prepareElementosMenu. Capturen la pantalla correspondiente.



1. **CONTROLADOR:** Construyan los controladores correspondientes a estas acciones. Para esto creen el método prepareAccionesMenu y los métodos base del controlador

( opcionSalver,opcionAbir, opcionExportar, opcionImportar, opcionIniciar, opcionCerrar), Estos últimos métodos, por ahora, sólo deben llaman directamente el método correspondiente de la capa de aplicación. Capturen una pantalla significativa.

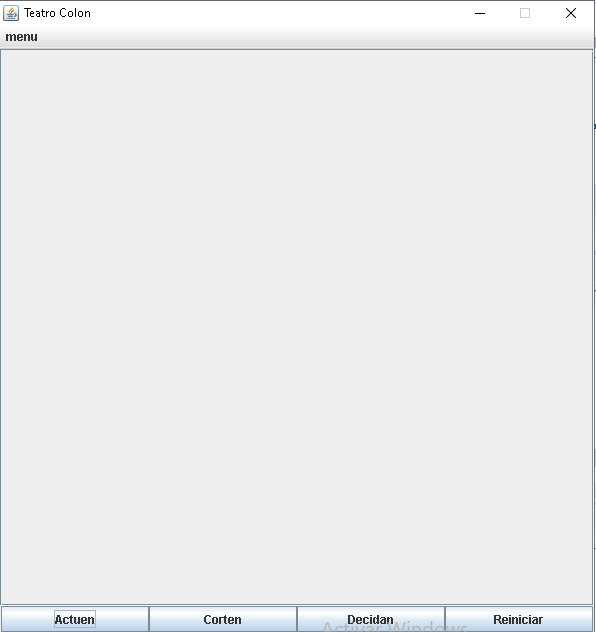


## Implementando salir e iniciar

**[En lab06.doc, \*.asta y \*.java] [NO OLVIDEN BDD y MDD]**

Las opciones salir e iniciar van a ofrecer los dos servicios estándar de las aplicaciones. El primero no requiere ir a capa de aplicación y el segundo sí.

1. Construyan el método opcionSalir que hace que se termine la aplicación. No es necesario incluir confirmación.
2. Construyan el método opcionIniciar que crea un nuevo teatroColon. Capturen una pantalla significativa.

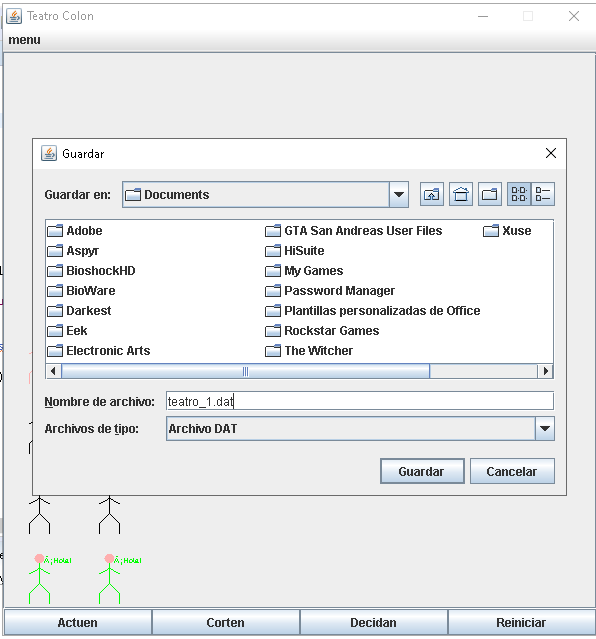


## Implementando salvar y abrir

**[En lab06.doc, \*.asta y \*.java] [NO OLVIDEN BDD y MDD]**

Las opciones salvar y abrir van a ofrecer servicios de persistencia de un teatroColon como objeto. Los nombres de los archivos deben tener como apellido .dat.

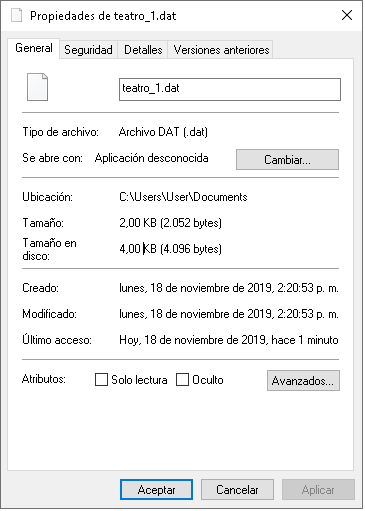
1. Construyan el método opcionSalvar que une de forma adecuada la capa de presentación con la capa de aplicación. Usen un FileChooser y atiendan la excepción. Ejecuten la aplicación probando las diferentes opciones del FileChooser y capturen una pantalla significativa.



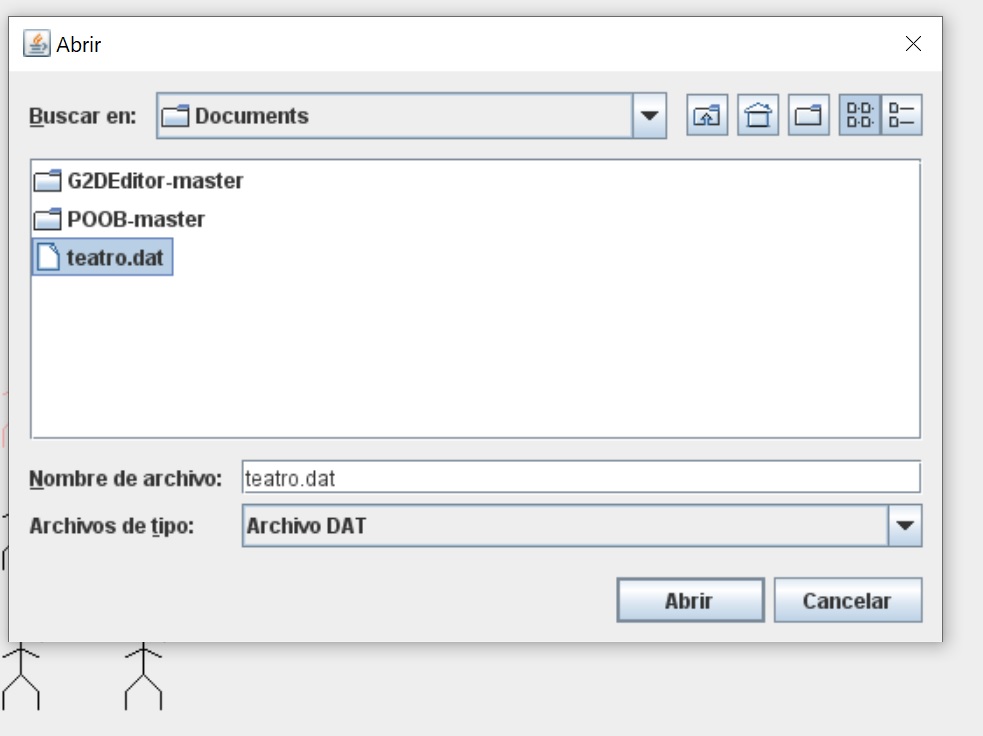
1. Construyan el método salve que ofrece el servicio de guardar en un archivo el estado actual del teatroColon.
2. Validen este método guardando el teatroColon inicial después de dos clics como

unteatroColon.dat. ¿El archivo se creó en el disco? ¿Cuánto espacio ocupa?

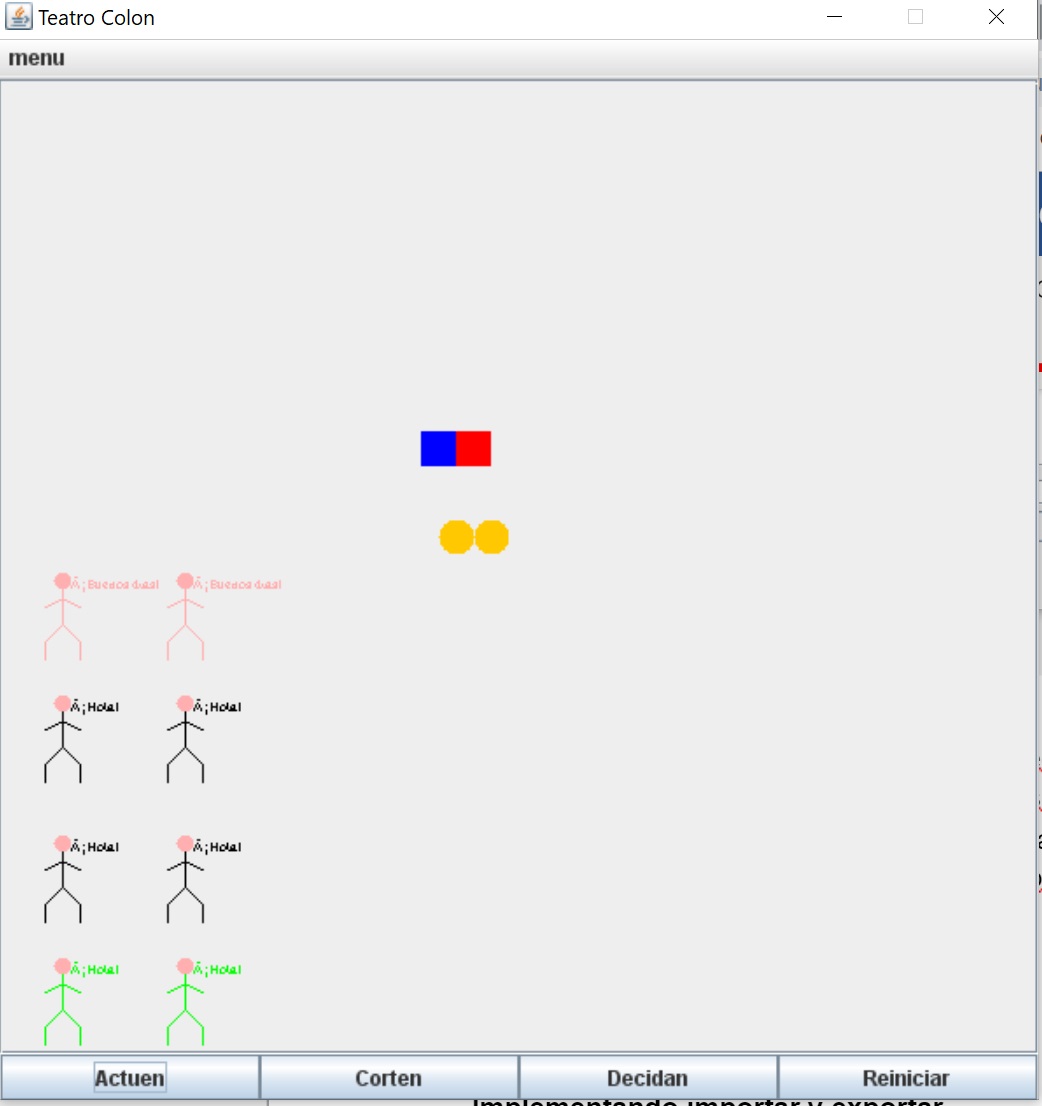
Sí.



1. Construyan el método opcionAbrir que une de forma adecuada la capa de presentación con la capa de aplicación. Ejecuten la aplicación probando las diferentes opciones del FileChooser y capturen una pantalla significativa.



1. Construyan el método abra que ofrece el servicio de leer un teatroColon de un archivo. Por ahora para las excepciones sólo consideren un mensaje de error general
2. Realicen una prueba de aceptación para este método iniciando la aplicación, creando una nueva situación en el teatroColon y abriendo el archivo unteatroColon.dat. Capturen imágenes significativas de estos resultados.



## Implementando importar y exportar

**[En lab06.doc, \*.asta y \*.java] [NO OLVIDEN BDD y MDD]**

Estas operaciones nos van a permitir importar información de un teatroColon desde un archivo de texto y exportarlo. Los nombres de los archivos de texto deben tener como apellido .txt

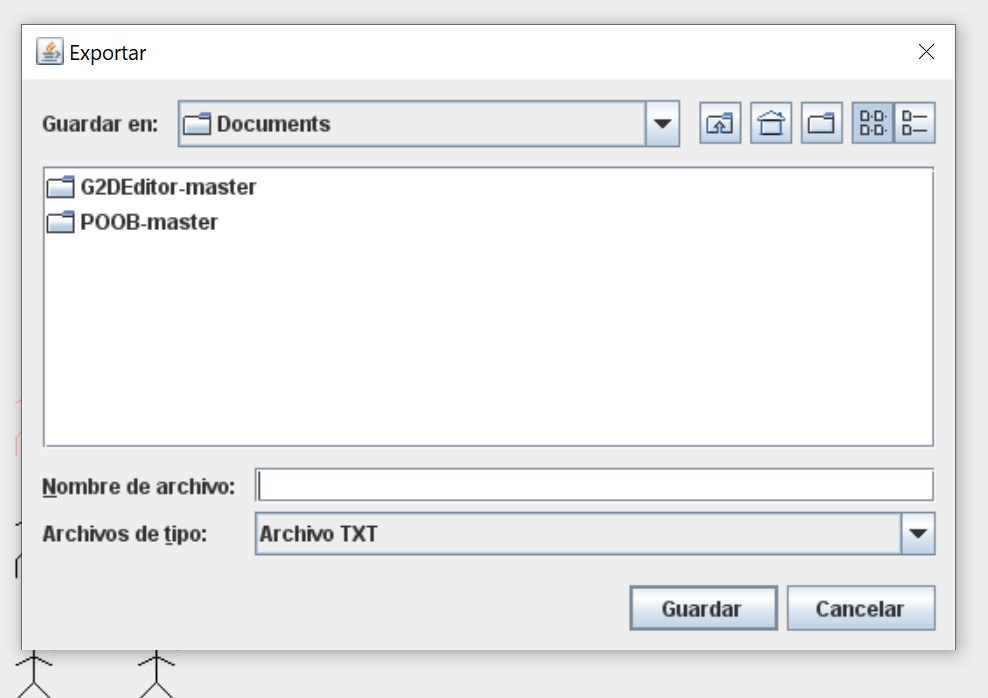
Los archivos texto tienen una línea de texto por cada elemento

En cada línea asociada un elemento se especifica el tipo y la posición.

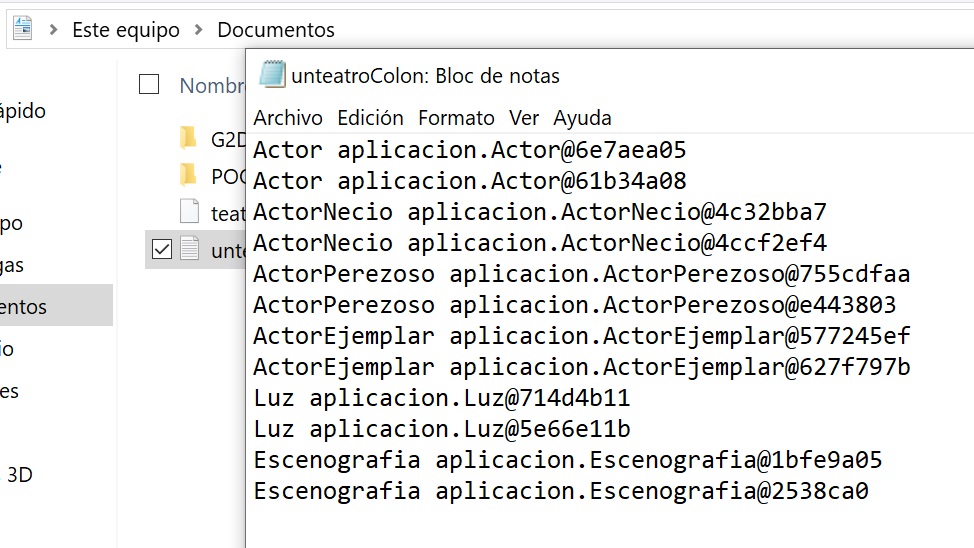
Normal 20 20 Romeo

Luz 50 50

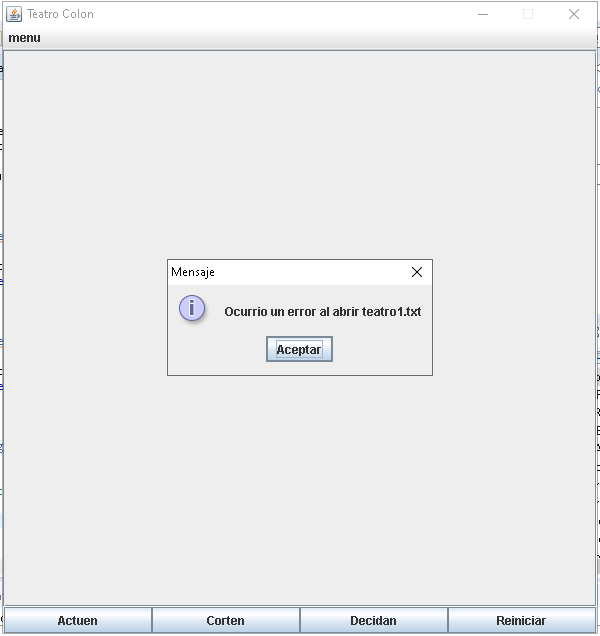
1. Construyan el método opcionExportar que une de forma adecuada la capa de presentación con la capa de aplicación. Ejecuten la aplicación y capturen una pantalla significativa.



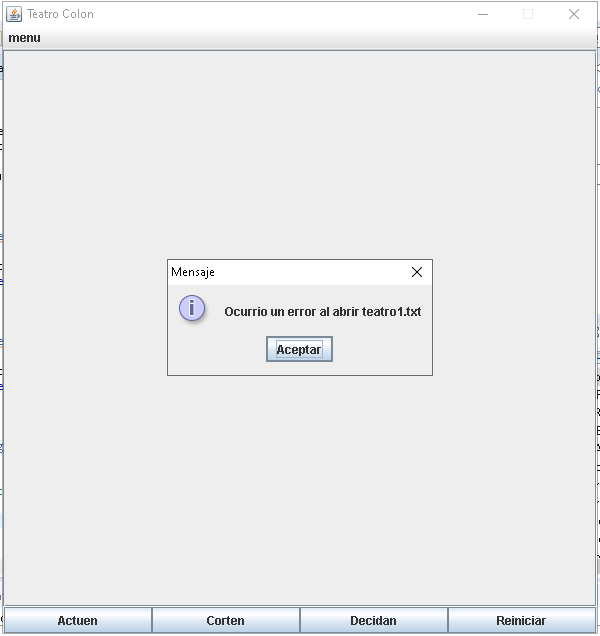
1. Construyan el método exporte que ofrece el servicio de exportar a un archivo texto, con el formato definido, el estado actual.
2. Realicen una prueba de aceptación de este método: iniciando la aplicación y exportando como unteatroColon.txt. Editen el archivo y analicen los resultados. ¿Qué pasó?



1. Construyan el método opcionImportar que une de forma adecuada la capa de presentación con la capa de aplicación. Ejecuten la aplicación y capturen una pantalla significativa.



1. Construyan el método importe que ofrece el servicio de importar de un archivo texto con el formato definido. Por ahora sólo considere un mensaje de error general. (Consulten en la clase String los métodos trim y split)
2. Realicen una prueba de aceptación de este par de métodos: iniciando la aplicación exportando a unteatroColon.txt. saliendo, entrando, creando un nuevo teatroColon e importando el archivo otroteatroColon.txt.. ¿Qué resultado obtuvieron? Capturen la pantalla final.



1. Realicen otra prueba de aceptación de este método escribiendo un archivo de texto

correcto en unteatroColon.txt. e importe este archivo. ¿Qué resultado obtuvieron? Capturen la pantalla.

## Analizando comportamiento

**[En lab06.doc, \*.asta y \*.java] [NO OLVIDEN BDD y MDD]**

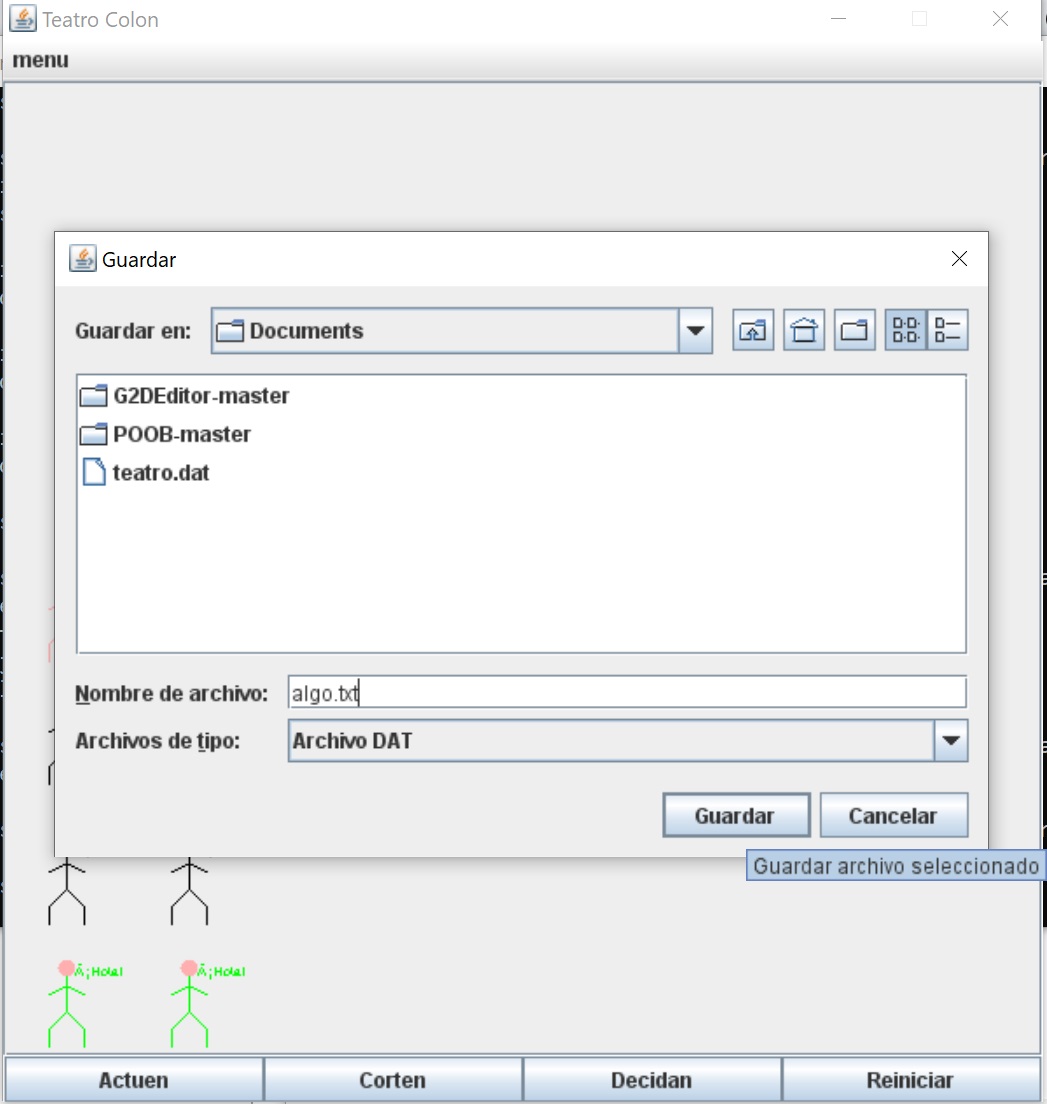
1. Ejecuten la aplicación, den tres clics, salven a un archivo cualquiera y ábranlo. Describan el comportamiento: Se guardó un archivo con extensión .dat con los objetos del teatro actualizados con los 3 comportamientos de los clics, y luego se generó un nuevo teatro tal cual como se guardó al abrir este mismo archivo.
2. Ejecuten la aplicación, tres clics, exporten a un archivo cualquiera e importen. Describan el comportamiento: Se guardó un archivo con extensión .txt con la informacion del teatro(los objetos), la cual volio a aparecer al importar el mismo archivo que exportamos.
3. ¿Qué diferencias ven el comportamiento 1. y 2.? Expliquen los resultados: En el comportamiento 1 se generó exactamente el mismo teatro que guardamos al abrirlo, pero en el comportamiento 2 no guardamos la informacion actualizada de las piernas y brazos en el txt entonces la posicon de los actores cambia(posicon de los brazos y de las piernas), solo guardamos la informacion más importante para nosotros al exportar, entonces al importar no quedó esta.

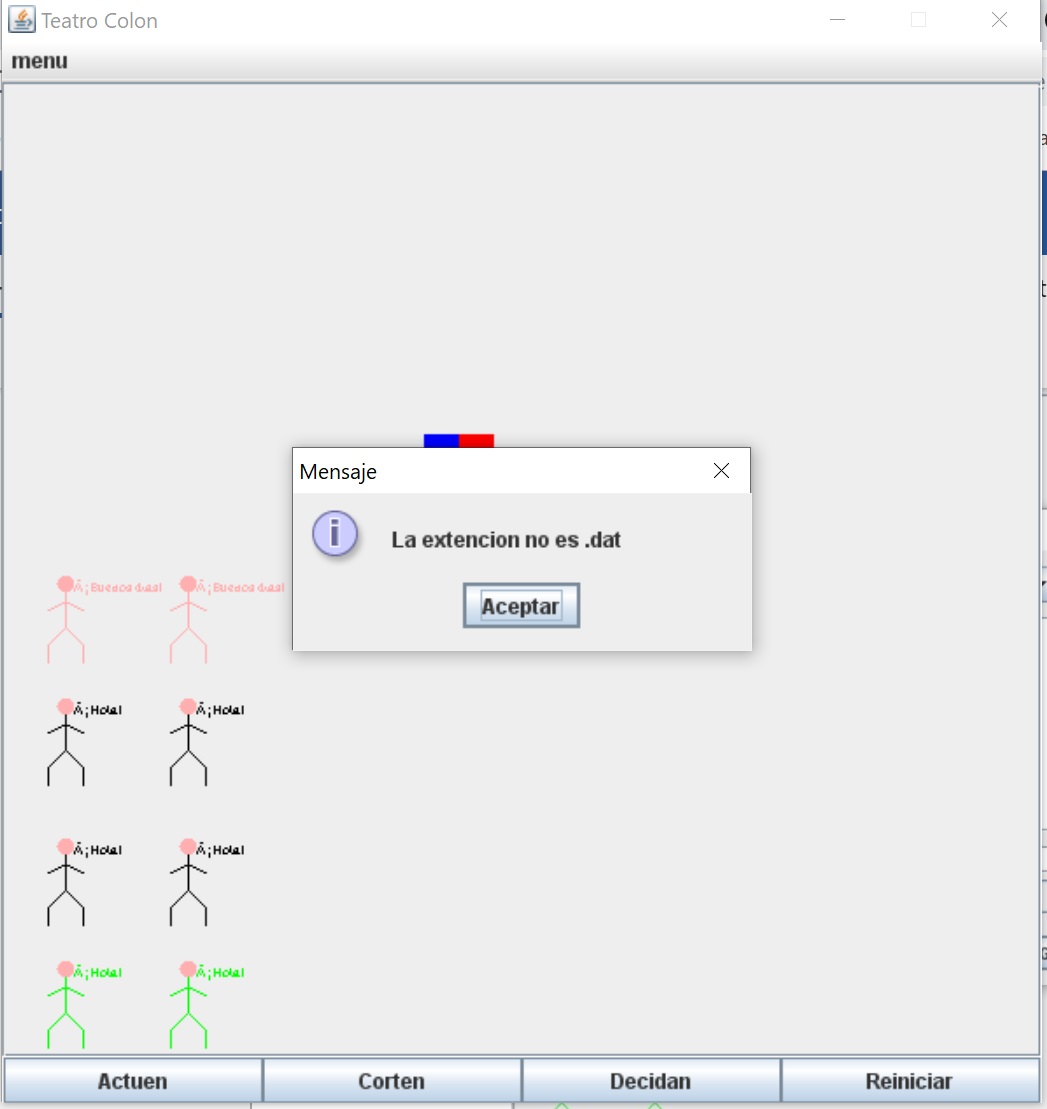
## Perfeccionando salvar y abrir

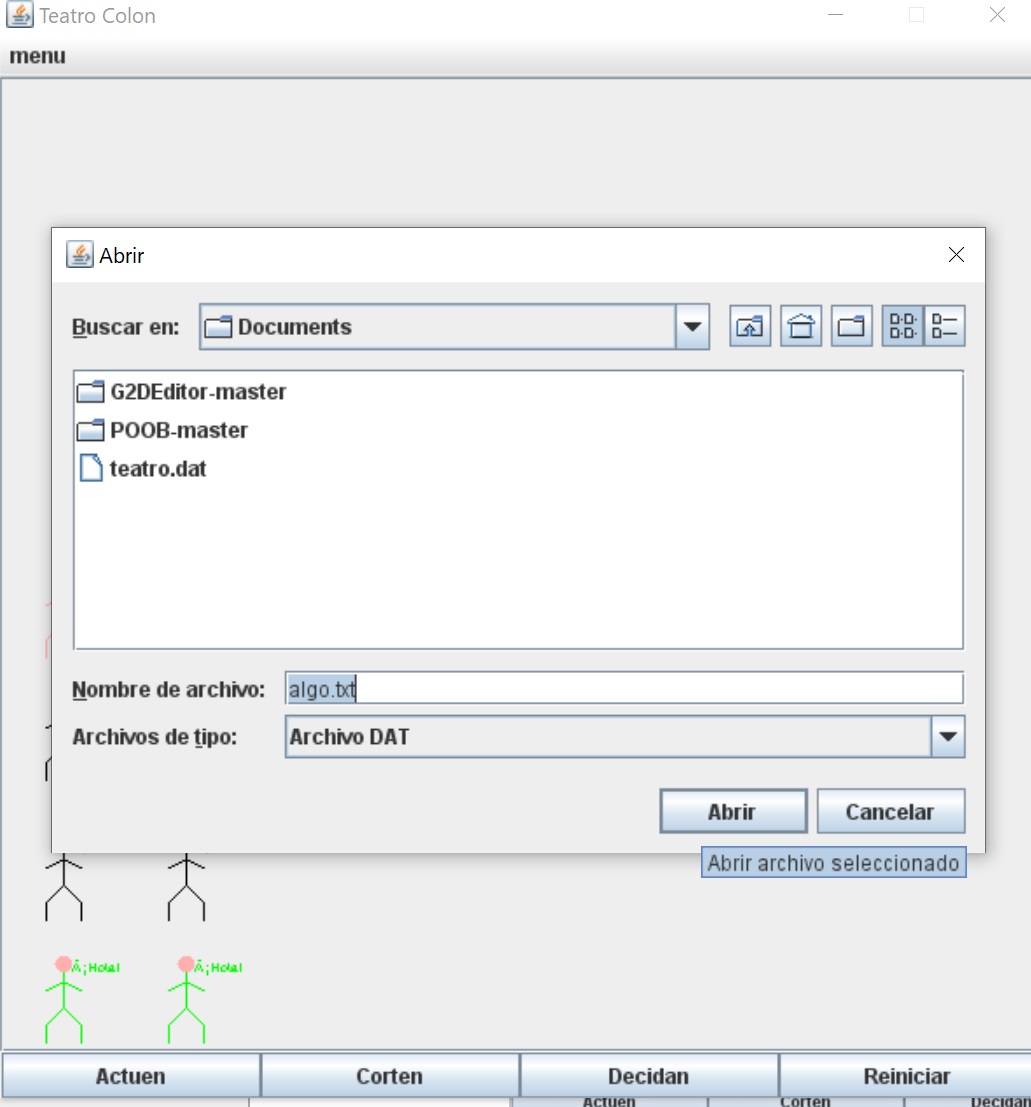
**[En lab06.doc, \*.asta y \*.java] [NO OLVIDEN BDD y MDD]**

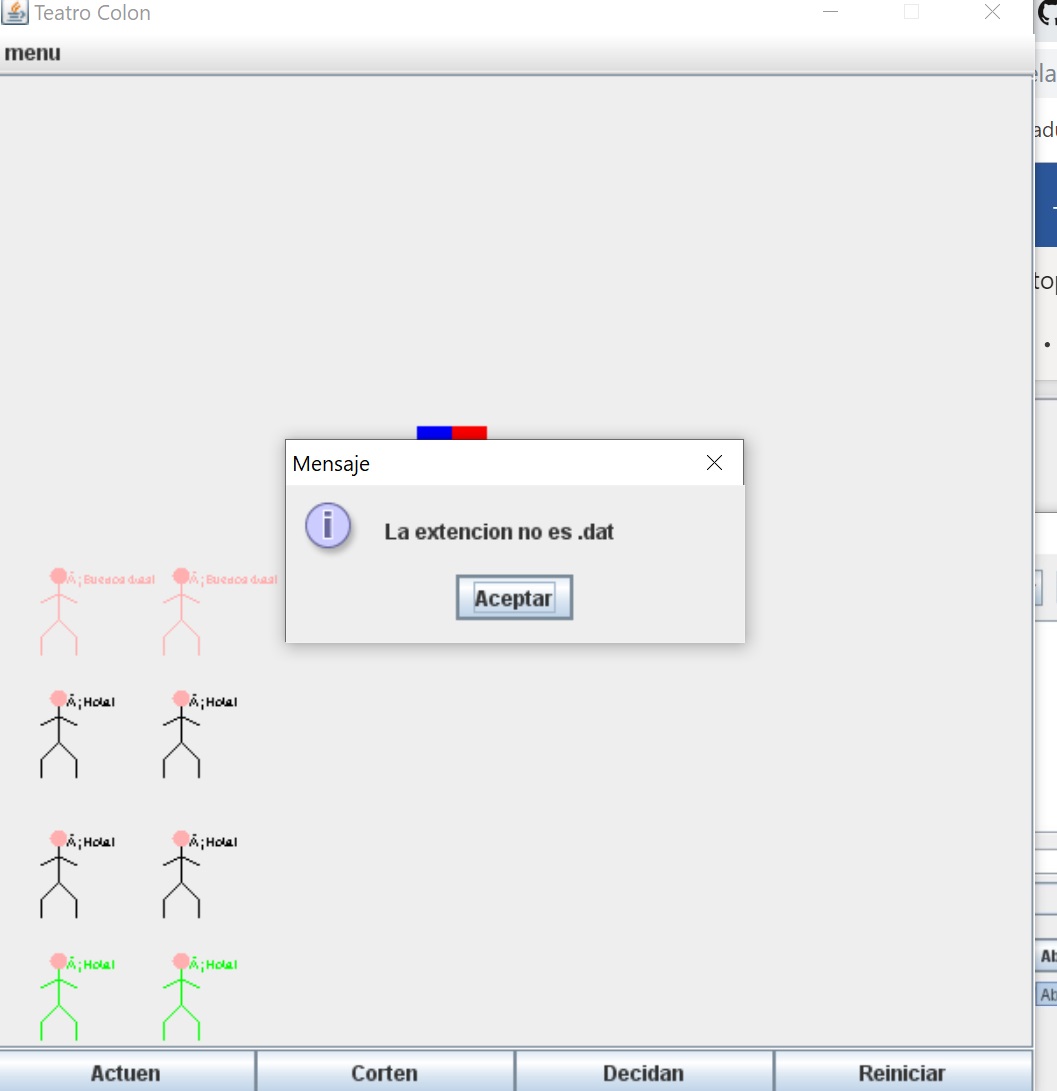
1. Copien las versiones actuales de abra y salve y renómbrenlos como abra01 y salve01
2. Perfeccionen el manejo de excepciones de los métodos abra y salve detallando los errores.
3. Realicen una prueba de aceptación para validar cada una de los nuevos mensajes

diseñados, ejecútenla y capturen la pantalla final









## 

## Perfeccionando importar y exportar.

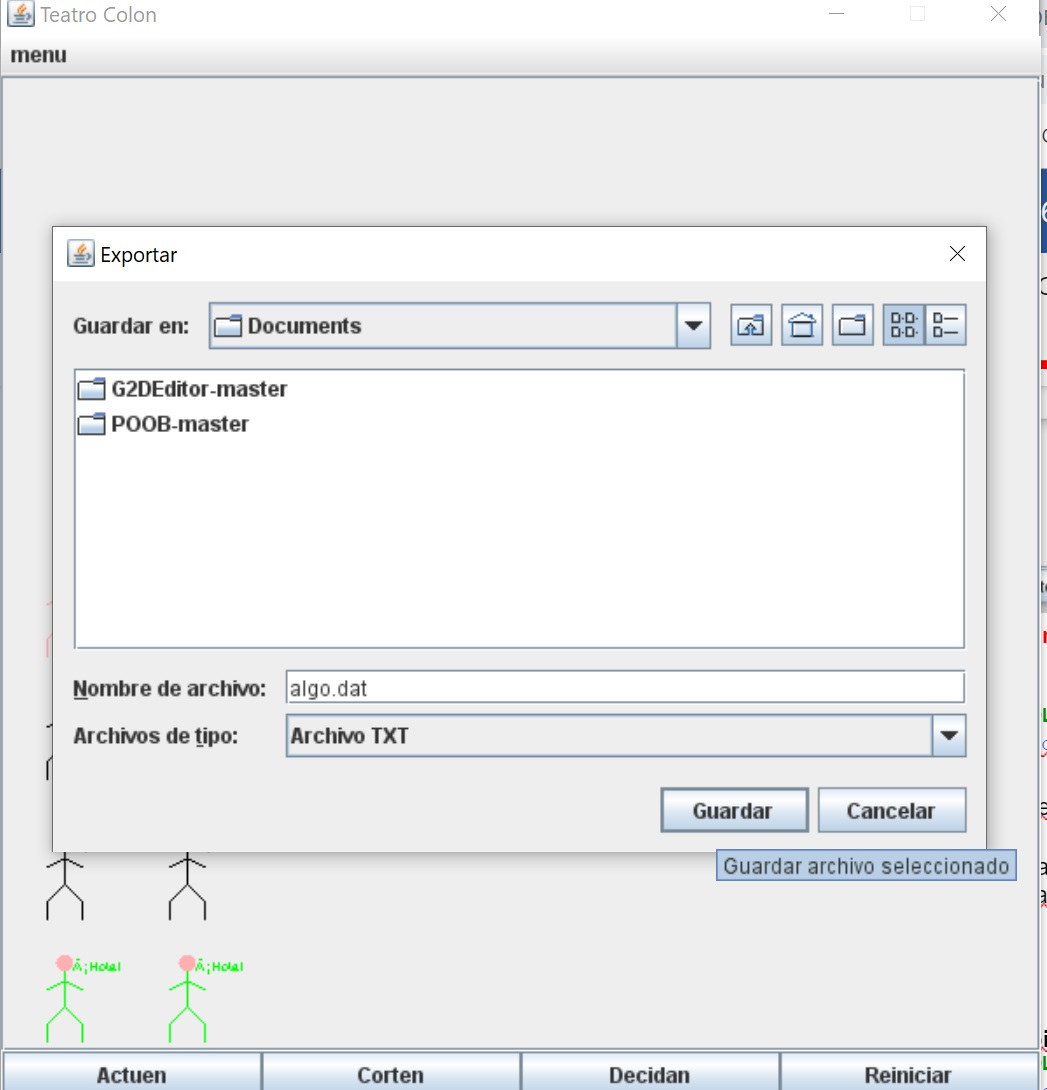
**[En lab06.doc, \*.asta , teatroColonErr.txt \*.java] [NO OLVIDEN BDD y MDD]**

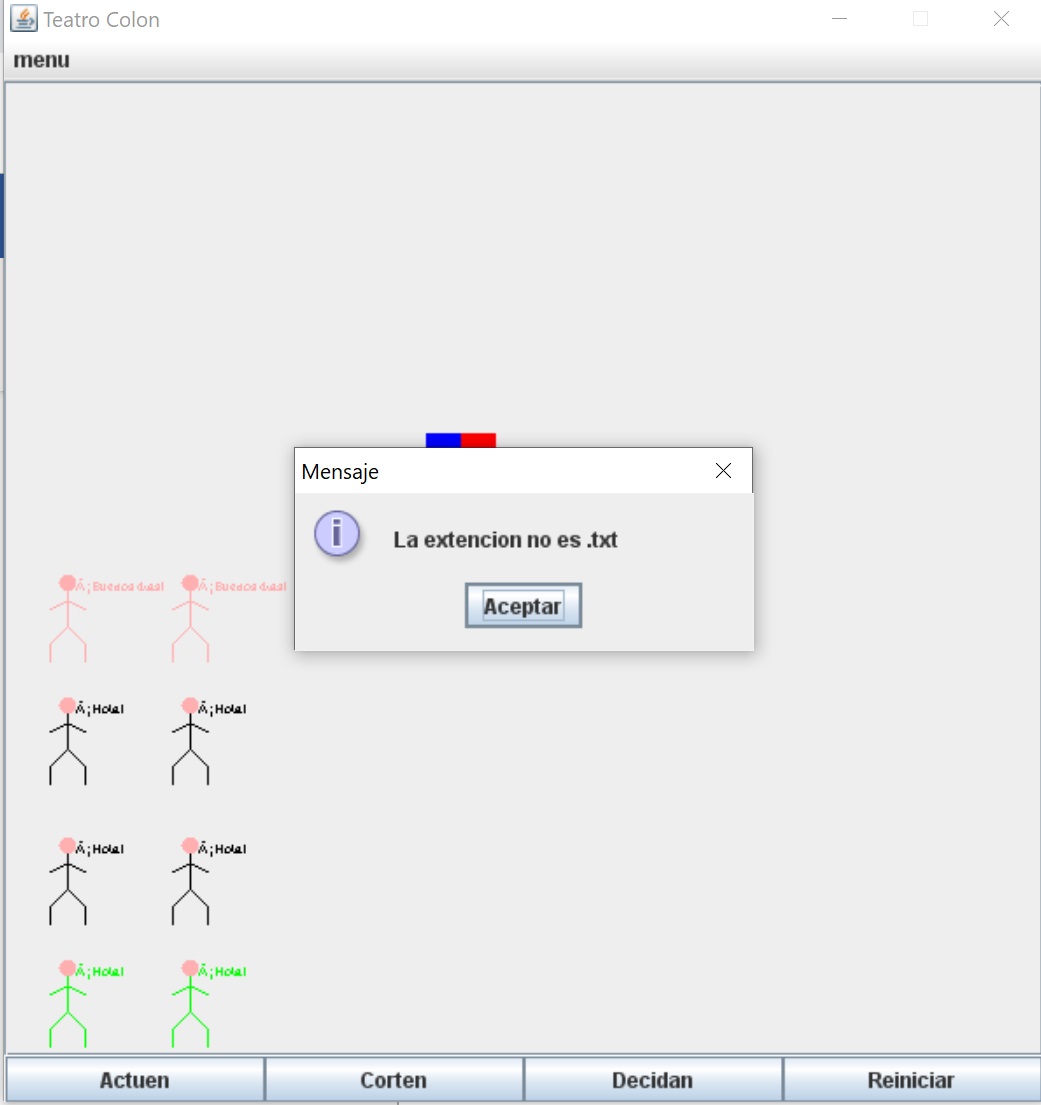
1. Copien las versiones actuales de importe y exporte y renómbrenlos como importe01

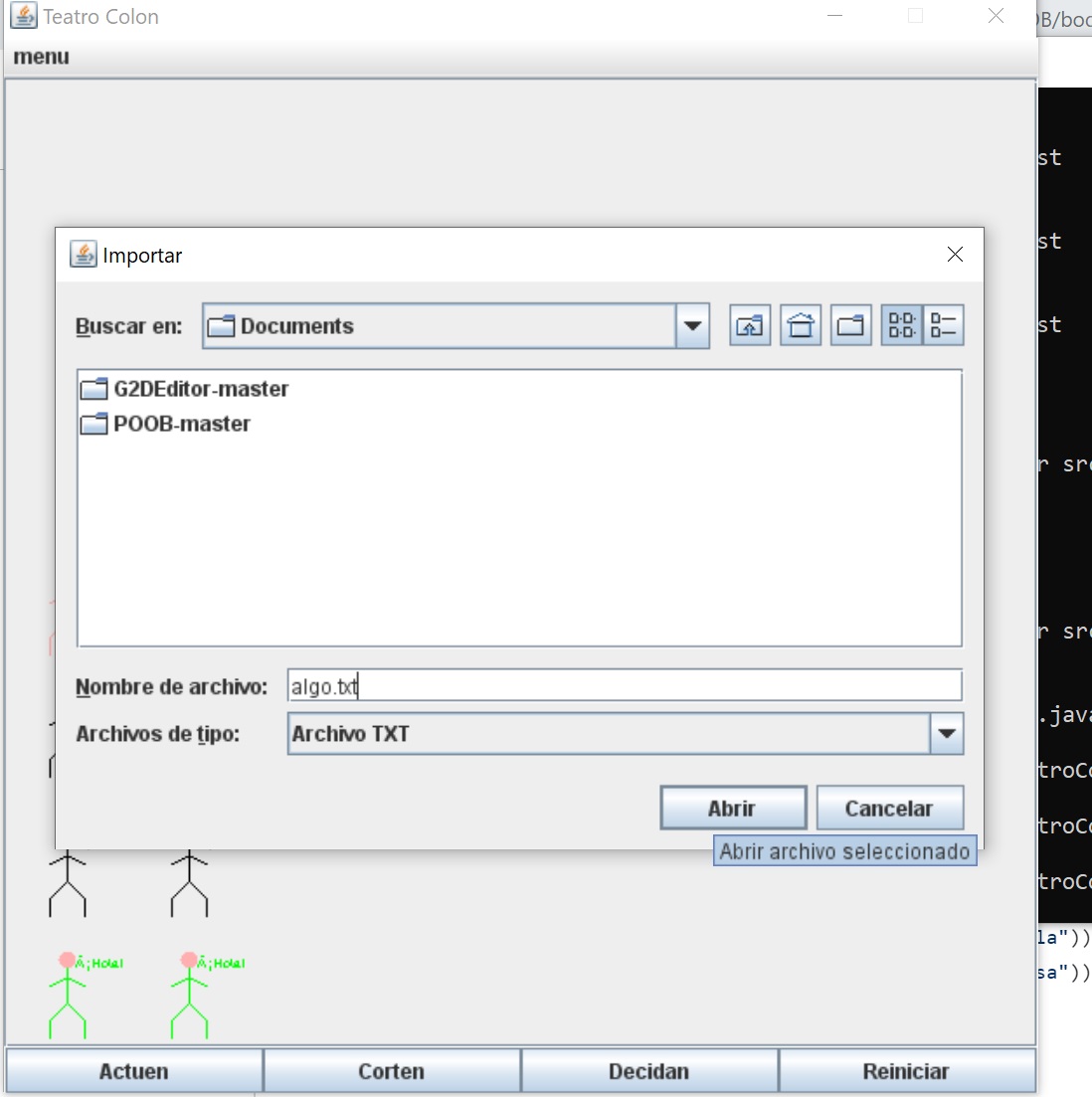
y exporte01

1. Perfeccionen el manejo de excepciones de los métodos importe y exporte detallando los errores.
2. Realicen una prueba de aceptación para validar cada una de los nuevos mensajes

diseñados, ejecútenla y capturen la pantalla final.







## Perfeccionando importar. Hacia un minicompilador.

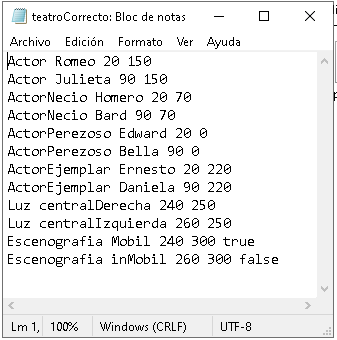
**[En lab06.doc, \*.asta , teatroColonErr.txt \*.java] [NO OLVIDEN BDD y MDD]**

1. Copien las versiones actuales de importe y exporte y renómbrenlos como importe02 y

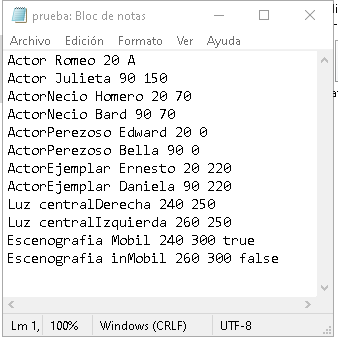
exporte02

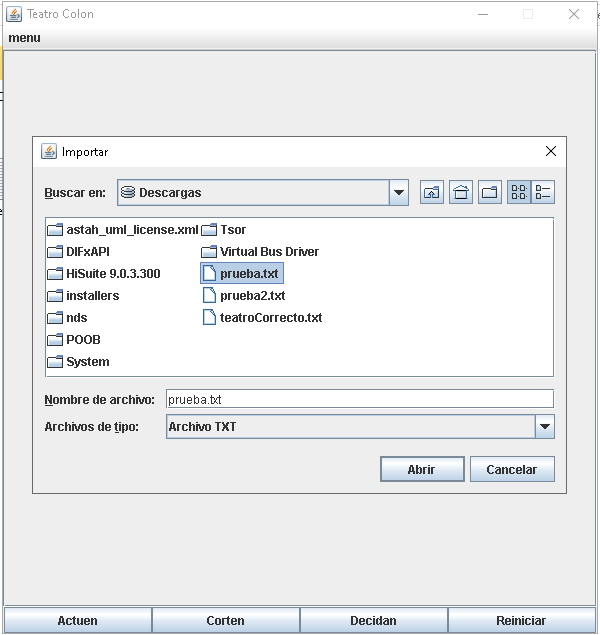
1. Perfeccionen el método **importe** para que, además de los errores generales, en las excepciones indique el detalle de los errores encontrados en el archivo (como un compilador) : número de línea donde se encontró el error, palabra que tiene el error y causa de error.
2. Escriban otro archivo con errores, llámelo teatroColonErr.txt, para ir arreglándolo con ayuda de su “importador”. Presente las pantallas que contengan los errores.

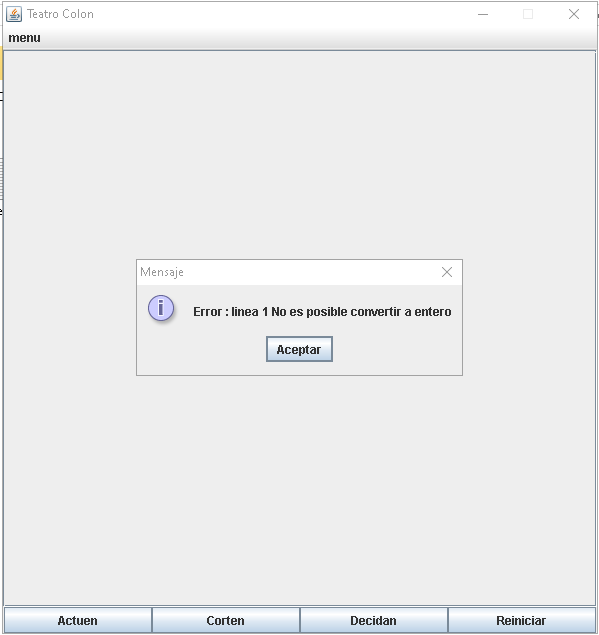
Primero un archivo que si funciona



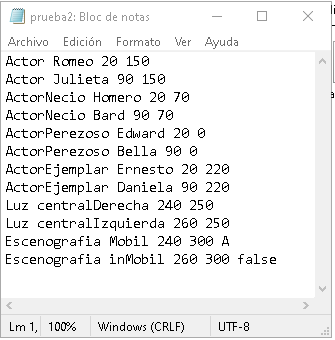
Ahora un archivo con un error en un numero

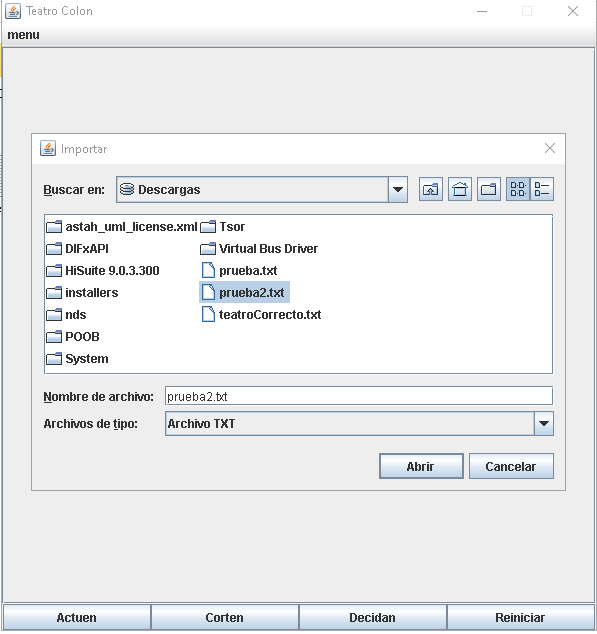


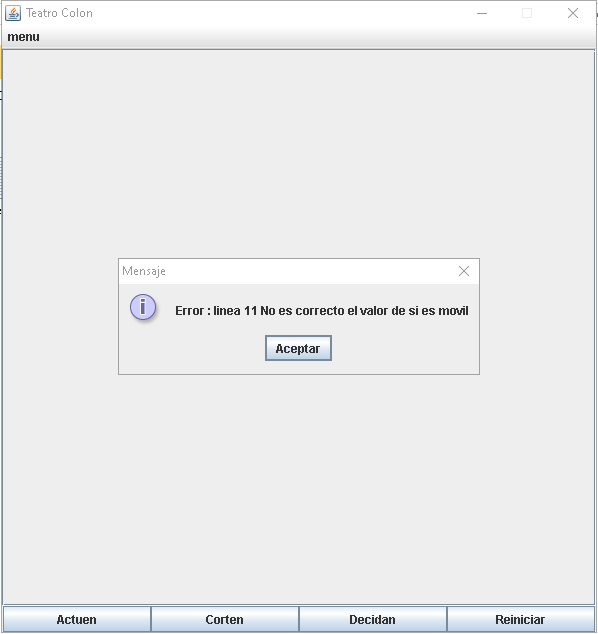




Ahora un archivo con errores en los Booleanos







## Perfeccionando importar. Hacia un minicompilador flexible.

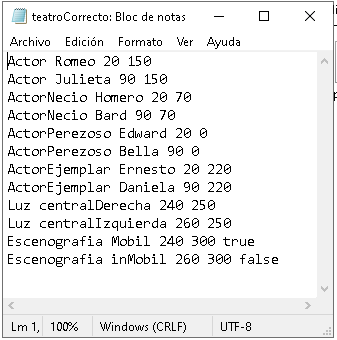
**[En lab06.doc, \*.asta , teatroColonFlex.txt \*.java] [NO OLVIDEN BDD y MDD]**

1. Copien las versiones actuales de importe y exporte y renómbrenlos como importe03 y

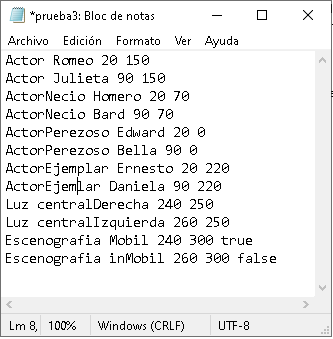
exporte03

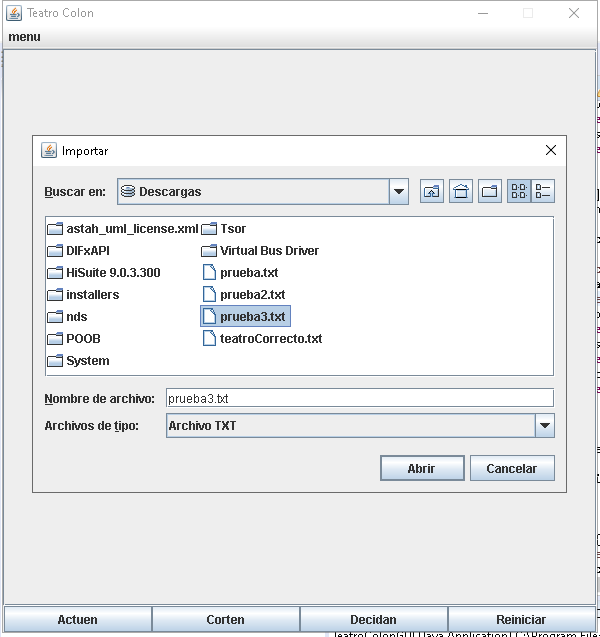
1. Perfeccionen los métodos importe y exporte para que pueda servir para cualquier tipo de elementos creados en el futuro (Investiguen cómo crear un objeto de una clase dado su nombre)
2. Escriban otro archivo de pruebas, llámelo teatroColonErrG.txt, para probar la flexibilidad. Presente las pantallas que contenga un error significativo.

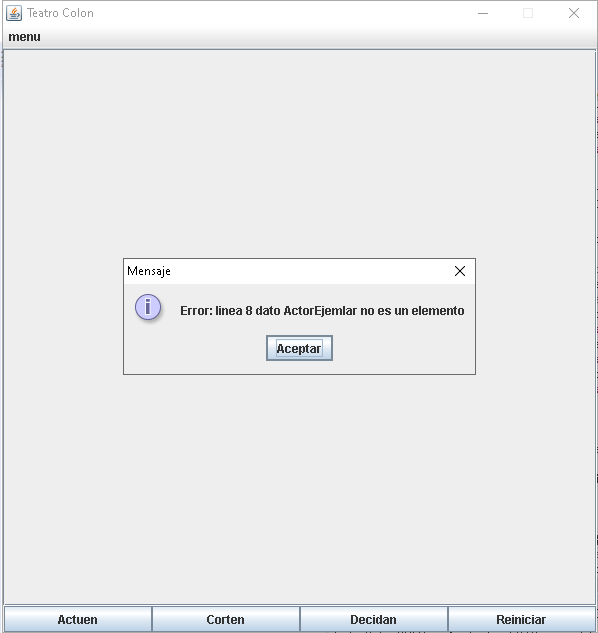
Primero un archivo que si funciona



El Archivo en mal estado







# RETROSPECTIVA

1. ¿Cuál fue el tiempo total invertido en el laboratorio por cada uno de ustedes

? 22 horas cada uno.

1. ¿Cuál es el estado actual del laboratorio? ¿Por qué? Completo..
2. Considerando las prácticas XP del laboratorio. ¿cuál fue la más útil? ¿por qué? Refactorizar cuando y donde sea posible, evita las redundancias y hace que el codigo sea más fácil de extender y más funcional. Todo el código debe pasar todas las pruebas unitarias antes de que se pueda liberar para asegurar que las funcionalidades son correctas y que se cubre el funcionamiento completo del codigo.
3. ¿Cuál consideran fue el mayor logro? ¿Por qué? Lograr aprender e implementar sobre entrada y salida, ya que no teniamos el conocimiento al respecto cuanto a la practica sobre el tema.
4. ¿Cuál consideran que fue el mayor problema técnico? ¿Qué hicieron para resolverlo? Lograr trabajar en conjunto a un ritmo rapido, porque no podiamos trabajar por separado o aumnetaban los errores. Trabajamos siempre en conjunto y ayudandonos entre los dos para tener una ultima buena practica y poder usarla de manera adecuada en el proyecto final.
5. ¿Qué hicieron bien como equipo? ¿Qué se comprometen a hacer para mejorar los resultados? Tener una buena comunicacion a la hora de diseñar y desarrollar codigo. Seguir teniendo una buena relacion de amistad y a la vez academica, para lograr siempre desarrollar los laboratorios de la mejor manera, y dedicarle el tiempo necesario.