**ESCUELA COLOMBIANA DE INGENIERÍA**

**PROGRAMACIÓN ORIENTADA A OBJETOS**

**LABORATORIO N°2**

* **Brayan Santiago Buitrago Murillo**
* **Cesar Eduardo Gonzalez Cuentas**

**Agosto 2019**

**Bogotá**

**PROGRAMACIÓN ORIENTADA A OBJETOS**

**Herencia e interfaces**

**ADEMAS Java desde consola**

**2019-02**

**Laboratorio 3/6**

**OBJETIVOS**

Desarrollar competencias básicas para:

1. Aprovechar los mecanismos de la herencia y el uso de interfaces.
2. Organizar las fuentes en paquetes.
3. Usar la utilidad jar de java para entregar una aplicación.
4. Extender una aplicación cumpliendo especificaciones de diseño, estándares y verificando su corrección.
5. Vivenciar las prácticas XP : The project is divided into [iterations](http://www.extremeprogramming.org/rules/iterative.html).
6. Utilizar los programas básicos de java (javac, java, javadoc, jar), desde la consola.

**ENTREGA**

 Incluyan en un archivo .zip los archivos correspondientes al laboratorio. El nombre debe ser los dos apellidos de los miembros del equipo ordenados alfabéticamente.

 En el foro de entrega deben indicar el estado de avance de su laboratorio y los problemas pendientes por resolver.

 Deben publicar el avance al final de la sesión y la versión definitiva en la fecha indicada en los espacios preparados para tal fin.

# DESARROLLO Contexto

En esta aplicación se presenta el escenario de un teatro, en nuestro caso el Teatro Colón.

**Conociendo [En lab03.doc y TeatroColon.asta ]**

1. **En el directorio descarguen los archivos contenidos en TeatroColon.zip. Revisen el código de la aplicación**
2. **¿Cuántos paquetes tiene?**

Tiene dos paquetes uno que es aplicación y la otra presentación.

1. **¿Cuántas clases tiene en total? ¿Cuántas tienen fuentes?**

6 clases en total y 5 tienen fuentes menos 1 que es la clase luz

1. **¿Cuál es la clase ejecutiva? ¿Por qué?**

La clase ejecutora es la clase teatroColonGUI ya que es la interfaz gráfica de la aplicación que utiliza las clases del paquete de aplicación y además contiene el método estático main que es el que se encarga de dar arranque a la interfaz

1. **Ejecuten el programa. ¿Qué funcionalidades ofrece? ¿Qué hace actualmente? ¿Por qué?** Actualmente el programa solo ofrece la funcionalidad de crear la interfaz del teatro, donde están los botones actúen, corten y decidan, pero no dejaba crear actores y los botones no funcionan.

**Arquitectura general. [En lab03.doc y TeatroColonasta]**

1. **Consulte el significado de las palabras package e import de java. ¿Qué es un paquete? ¿Para qué sirve?**

**Package:** esta instrucción se encarga de declarar que la clase en la que se ejecuta esa instrucción pertenece al paquete al cual se indica

**Import:** esta instrucción nos permite acceder a otras clases de otros paquetes de esta forma se pueden usar más clases así no estén implementadas en el paquete en el que se está trabajando.

1. **Revise el contenido del directorio de trabajo y sus subdirectorios. Describa su contenido. ¿Qué coincidencia hay entre paquetes y directorios?**

Hay tres directorios uno de documentación del Proyecto que contiene archivos HTML con la documentación de todo el Proyecto, además contiene dos directorios con el mismo nombre de los paquetes y cada uno contiene archivos.class que son el código de las clases del paquete, archivos.java que son los ejecutables de las clases del paquete y archivos.txt que son los respectivos comentarios de las clases del paquete

1. **Inicie el diseño con un diagrama de paquetes en el que se presente los componentes y las relaciones entre ellos.**

**En astah, crear un diagrama de clases (cambiar el nombre por Package**

**Diagram0)**

**Arquitectura detallada. [En lab03.doc y TeatroColonasta]**

1. **Usando ingeniería reversa prepararen el proyecto para MDD. Presente el diseño estructural actual de la aplicación (diagrama de clases). Las clases de la capa de presentación sólo deben tener los elementos públicos.**
2. **Adicione en las fuentes la clase de pruebas necesaria para BDD. (No lo adicione al diagrama de clases) ¿En qué paquete debe estar? ¿Por qué? ¿Asociado a qué clase? ¿Por qué?**

# Ciclo 1. Actuan y descansan los actores normales [En lab04.doc y \*.java]

**(NO OLVIDE BDD - MDD)**

1. Estudie la clase Teatro. ¿Qué tipo de colección se usa para albergar los elementos?

Los elementos se encuentran guardados en un arraylist que contiene objetos de tipo EnEscena

¿Puede recibir actores? ¿Por qué?

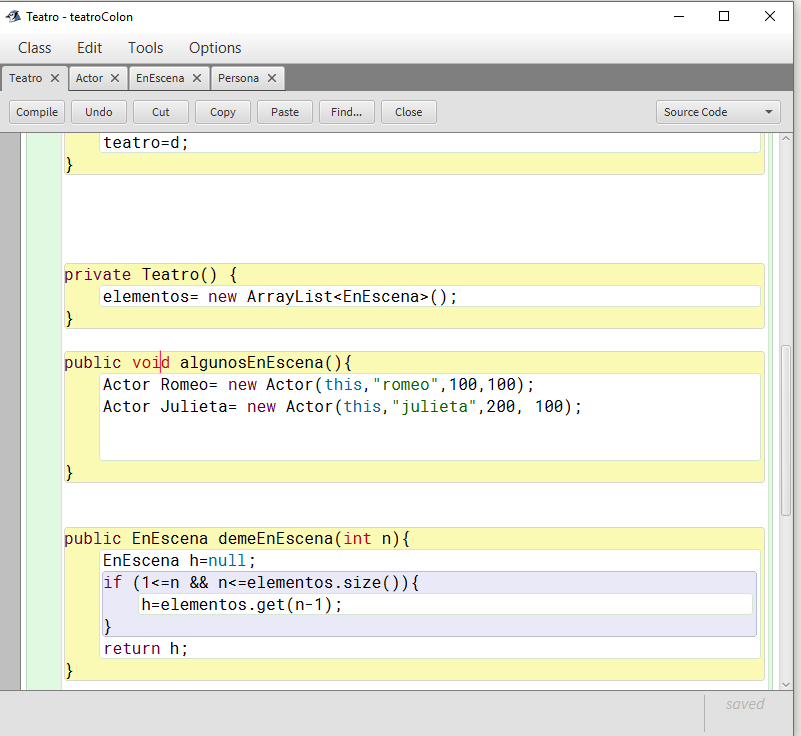
No lo recibe, ya que teatro es atributo de Actor, por ende es lo contrario Actor recibe a teatros

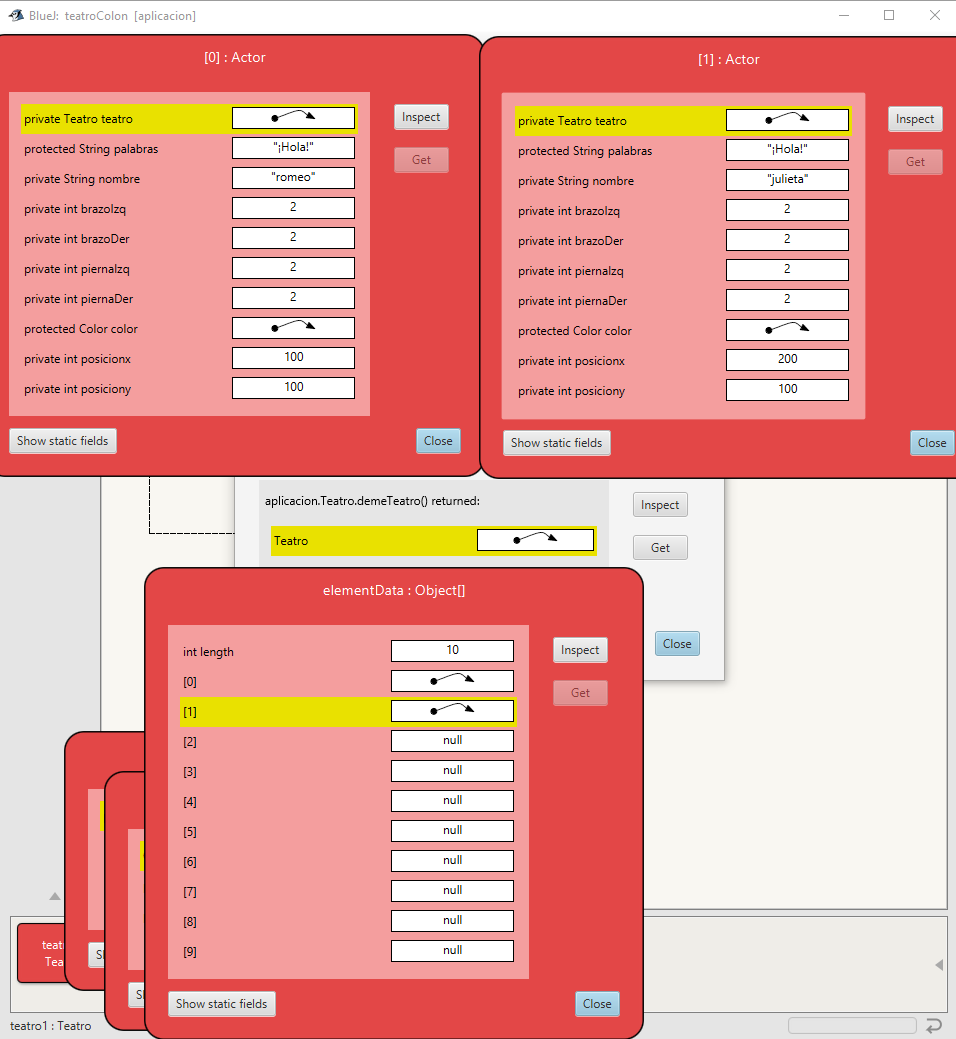
1. Estudie el código de la clase Actor ; ¿de qué color es? ¿qué palabras dice? ¿cómo entran en acción? ¿qué hacen cuando corta? ¿cómo deciden?

Es Amarillo, algo distinto de las otras clases es que extiende de persona y Tambien implementa la clase de interfaz EnPersona, cuando corta llama funciones de persona que son los movimientos de sus extremidades, todo se decide desde EnEscena porque ahi se presenta el metodo decida donde se mira si se corta o se actua

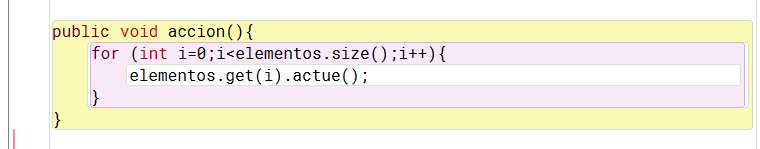
1. En el método algunosEnEscena de la clase Teatro cree dos actores en diferentes posiciones y acondiciónelos al Teatro llámelos romeo y julieta. Ejecute el programa y capture la pantalla. ¿Qué pasa ahora? ¿Pidales que entren en acción? ¿Qué pasa? ¿Por qué?

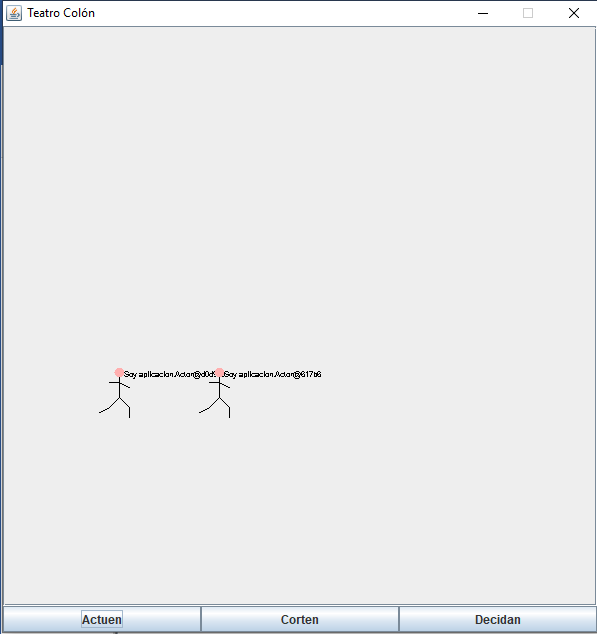
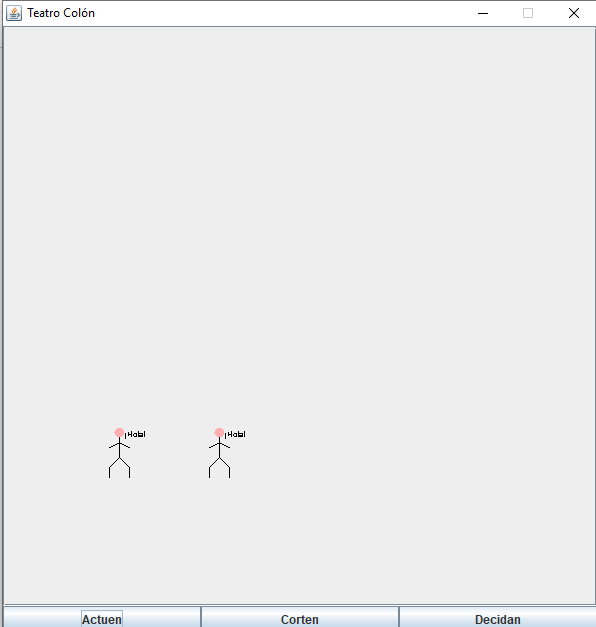
No pasa nada ya que accion no esta definido, pero se crean normal los dos actores



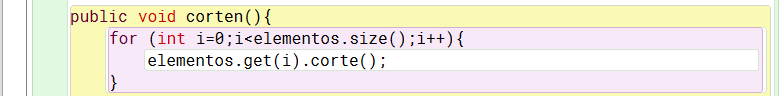


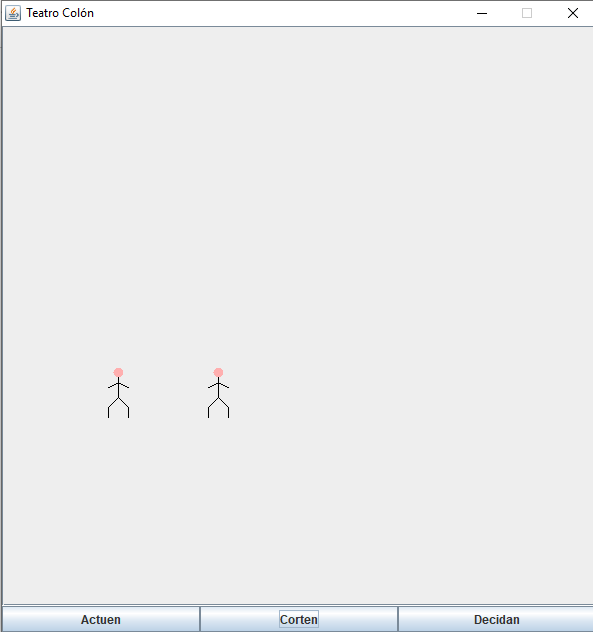
1. En este punto vamos a construir (diseño y código) el método que atiende el click del botón ***Accion*** de la interfaz: el método llamado accion() de la clase Teatro. Ejecute el programa y haga tres click en el botón **Accion**. ¿Cómo actúan romeo y julieta? Capture la pantalla inicial y la final.





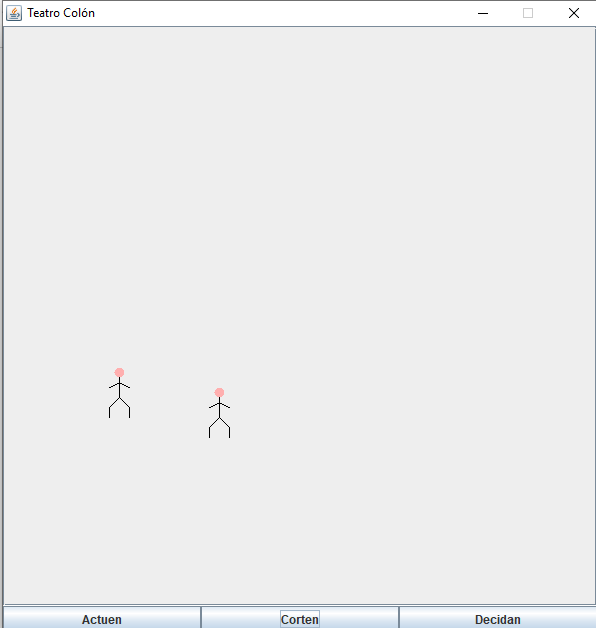
1. En este punto vamos a construir (diseño y código) el método que atiende el click del botón **corten** de la interfaz: el método llamado corten() de la clase Teatro. Construya el método, ejecute el programa y haga click en el botón **Corten**. ¿Como quedan todos los actores después de esta orden? Capture la pantalla inicial y la final.





1. En este punto vamos a construir (diseño y código) el método que atiende el click del botón **decidan** de la interfaz: el método llamado decidan() de la clase Teatro. Construya el método, ejecute el programa y haga click en el botón **Decidan**. ¿Como quedan todos los actores después de esta orden? Capture la pantalla inicial y la final.

Despues de dar decidan no pasa nada, pero cuando se actua y se corta despues de que se decide alguno de los dos sube una pocision en Y



**Ciclo 2. Incluyendo a los actores necios [En lab04.doc, TeatroColon.asta y \*.java]**

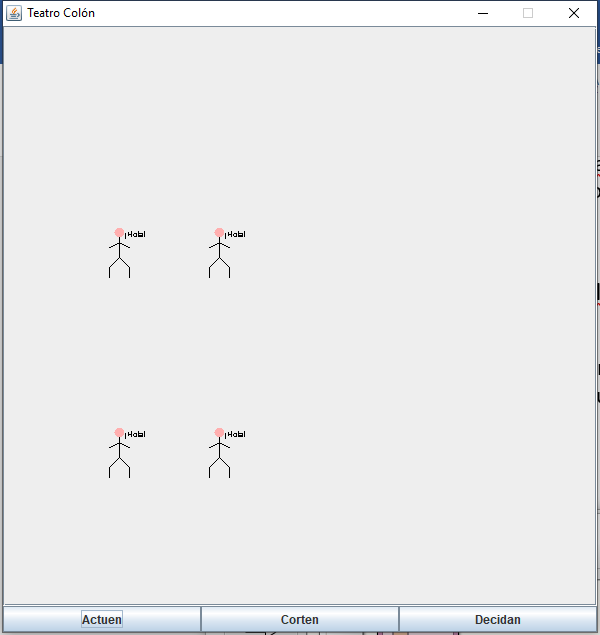
El objetivo de este punto es permitir recibir en el salón actores necios.

(NO OLVIDE BDD - MDD)

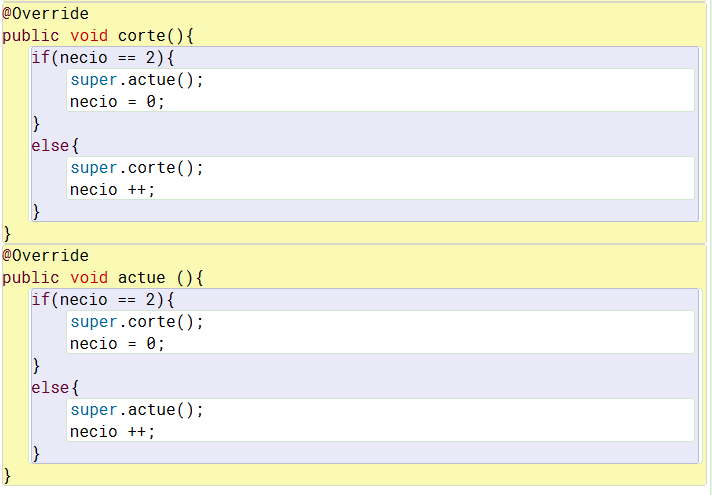
1. Los actores necios normalmente llevan la contraria. Si se les pide acción cortan y si se les pide que corten entran en acción. Adicionalmente, cuando les piden decidir hacen lo que hicieron la última vez. Implemente este nuevo actor. ¿cuáles métodos se sobre-escriben *(overriding)*?

Se crea la clase Actores necios, de donde se sobre escriben los metodos corten y actuen, hacienda que uno llame al otro y viceversa; esta nueva clase se extiende de Actor

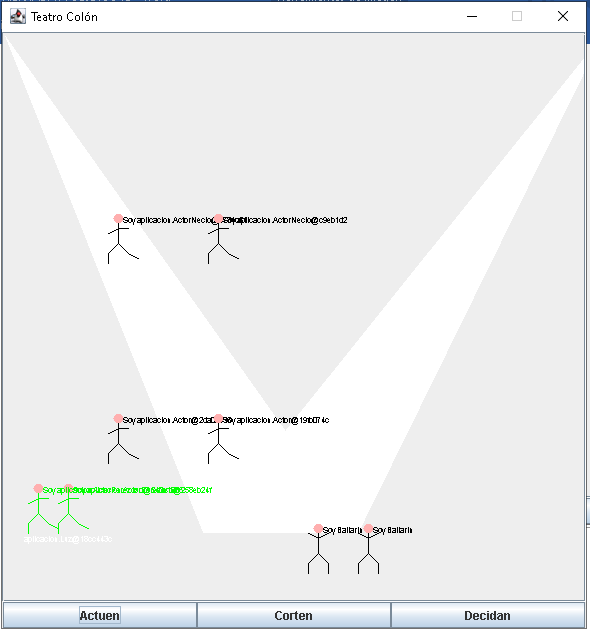
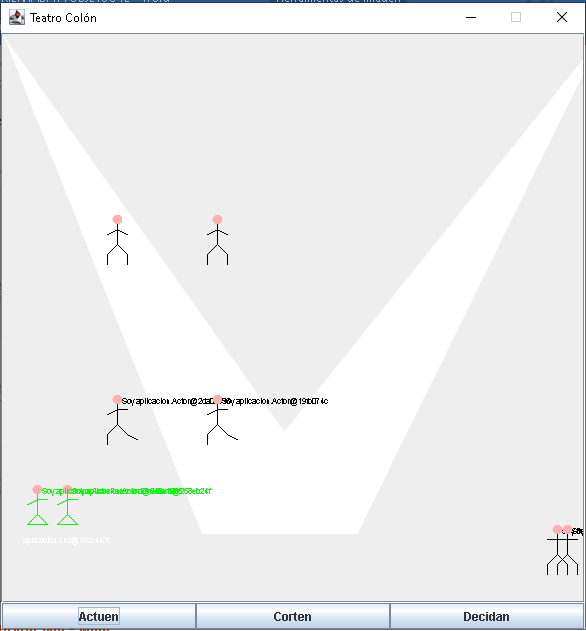
1. Adicione una pareja de actores necios, llámelos homer y bard, ejecute el programa y pídales a todos que actúen y que descansen. Capture la pantalla. ¿Qué pasa?

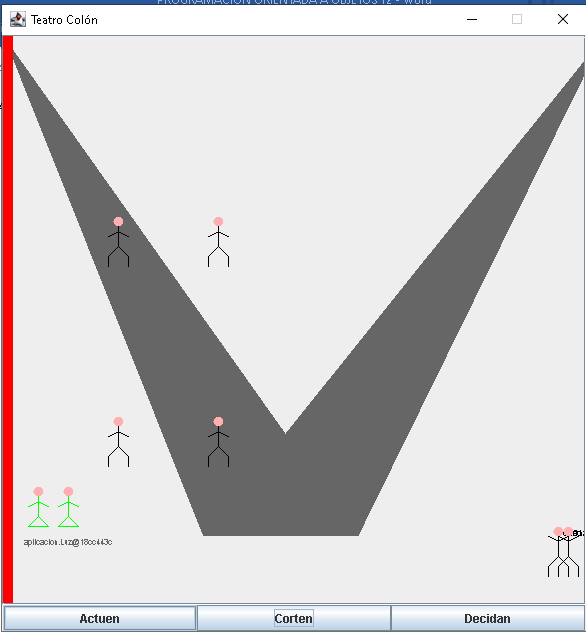


1. Ahora, los necios quieren sorprender con su necedad; es decir, sólo son necios cada tres veces. ¿Qué modificaría para lograr este comportamiento? ¡Hágalo!



1. Nuevamente ejecute el programa y pídales a todos que entren en acción y corten.

Capture la pantalla. ¿Qué pasa? Hicimos tres capturas para evidenciar las diferencias



**Ciclo 3. Adicionando luces** [En lab03.doc, TeatroColon.asta y \*.java]

El objetivo de este punto es incluir en el Teatro luces (sólo vamos a permitir este tipo de luces). Las luces se prenden cuando hay acción y se apagan cuando hay corte.

**(NO OLVIDE BDD – MDD)**

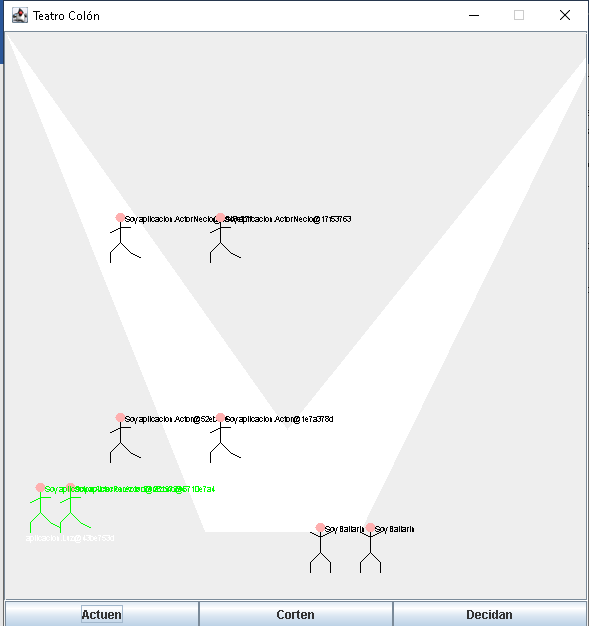
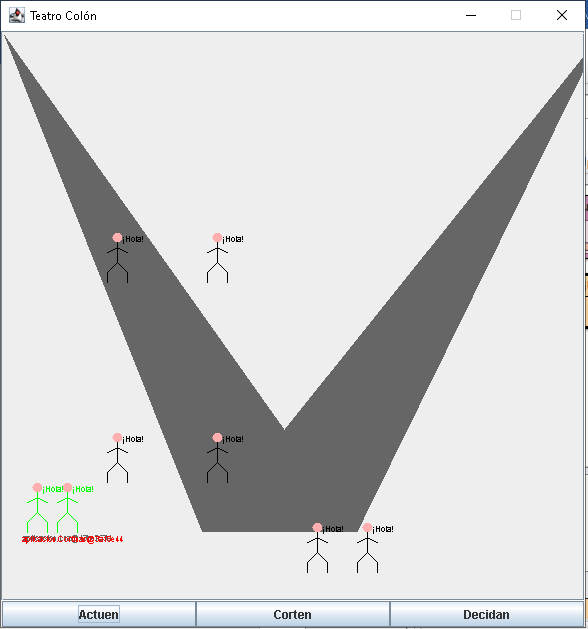
1. Construya la clase Luz para poder adicionarla en el Teatro. ¿qué cambios incluyó

Trabajamos sobre la clase luz y esta implementa EnEscena.

1. Para aceptar este elemento, ¿debe cambiar en el código del Teatro. en algo? ¿por qué?

En teatro solo se modificó código en AlgunosEnEscena.

1. Adicionen dos luces en el centro del Teatro, llámenlas centralDerecha y centralIzquierda, ejecuten el programa. ¿Qué pasa? ¿es correcto?

Es correcto, se pone de color gris cuando “corta” y cuando es “actué” se pone blanca simulando la luz

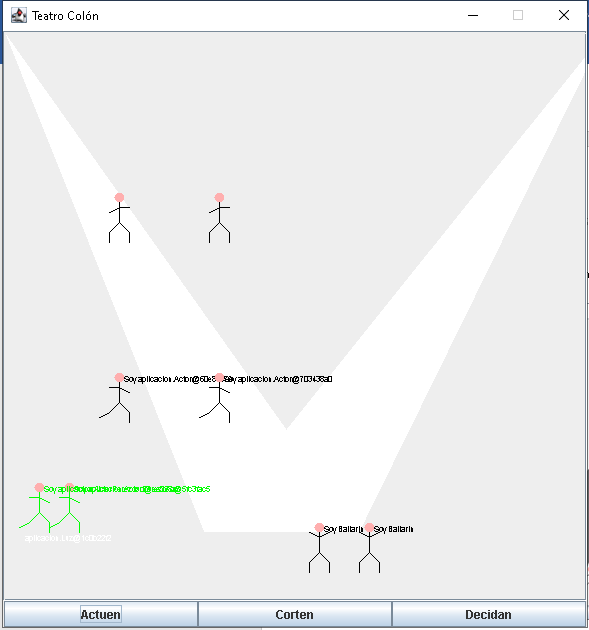
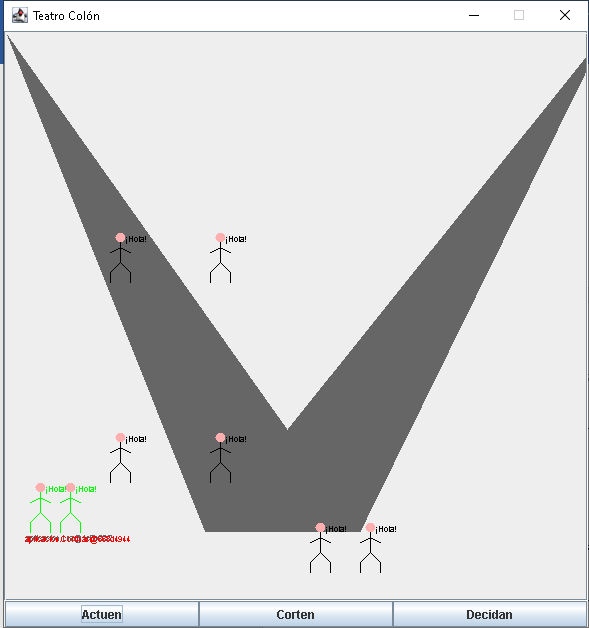
**Ciclo 4. Creando un nuevo actor: el perezoso**

El objetivo de este punto es incluir el actor perezoso, considerando que: Está vestido con color verde

Para implementar este actor, ¿qué cambios debería hacer al diagrama de clases? ¿a los de secuencia? Explique.

Crear una nueva clase que herede de la clase actor ya que comparten muchos atributos y métodos y los de secuencia son iguales a los de actor normal solo que cambia el nombre del método principal.

Implemente al actor Perezoso

Ahora sí, adicione ahora una pareja de actores perezosos llámelos bella y edward, ejecute el programa y haga tres click en el botón. ¿Cómo quedan todos los actores? Capture la pantalla inicial y la final.

# Ciclo 5. Nuevo actor: Proponiendo y diseñando

El objetivo de este punto es permitir recibir en un nuevo tipo de actor.

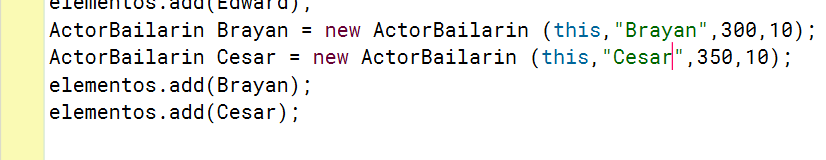
(NO OLVIDE BDD - MDD)

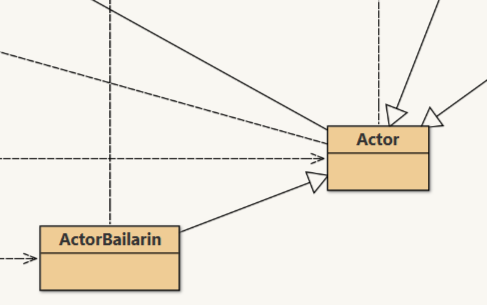
1. Propongan, describan e implementen un nuevo tipo de actor.

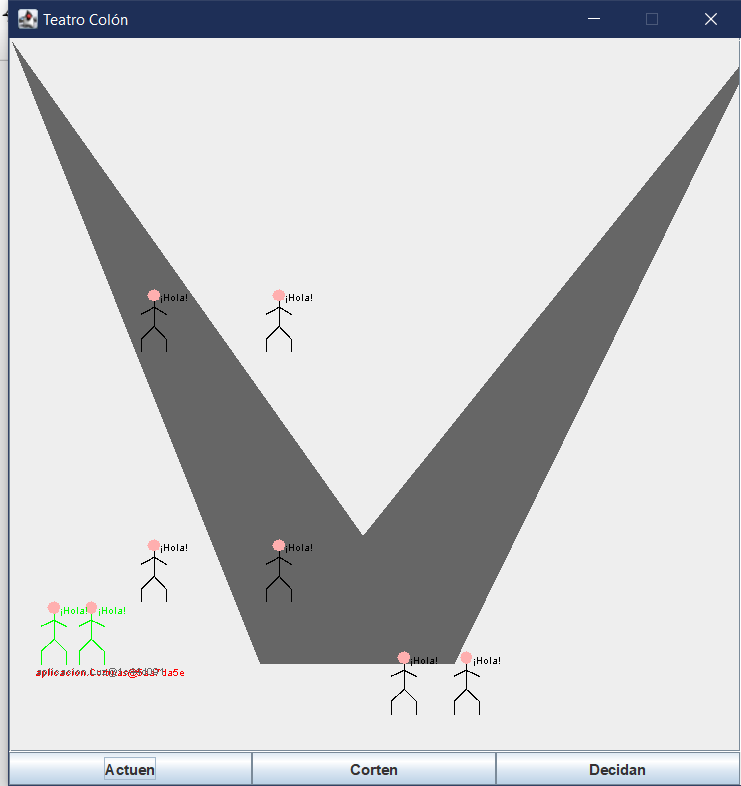
Actor Bailarin quien nunca baja los brazos solo los tiene al frente y arriba, y despues de 5 actuen salta volviendo a su pocision en Y, cuando se corta salen del scenario haciendose en el extremo

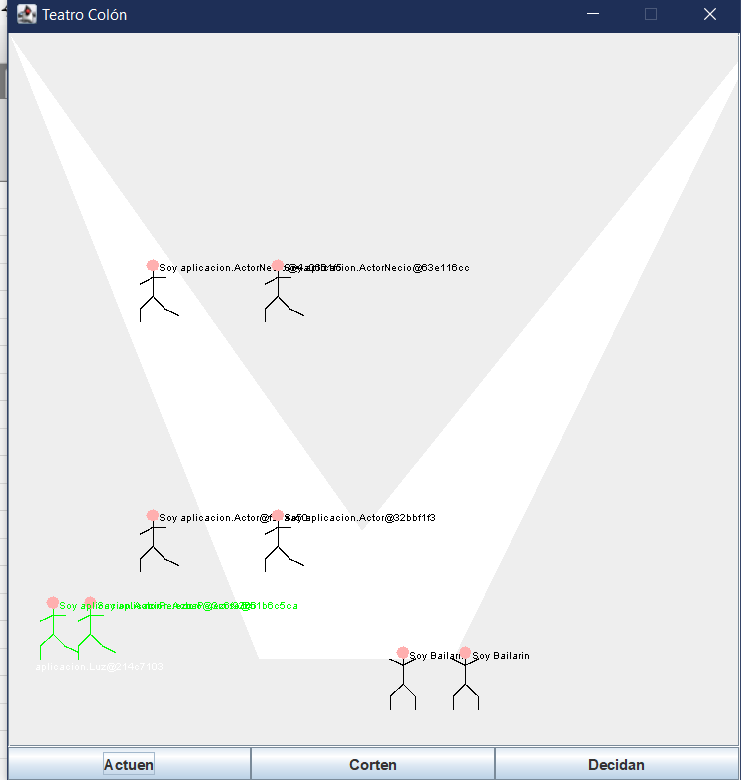
Y si decide cambia de color

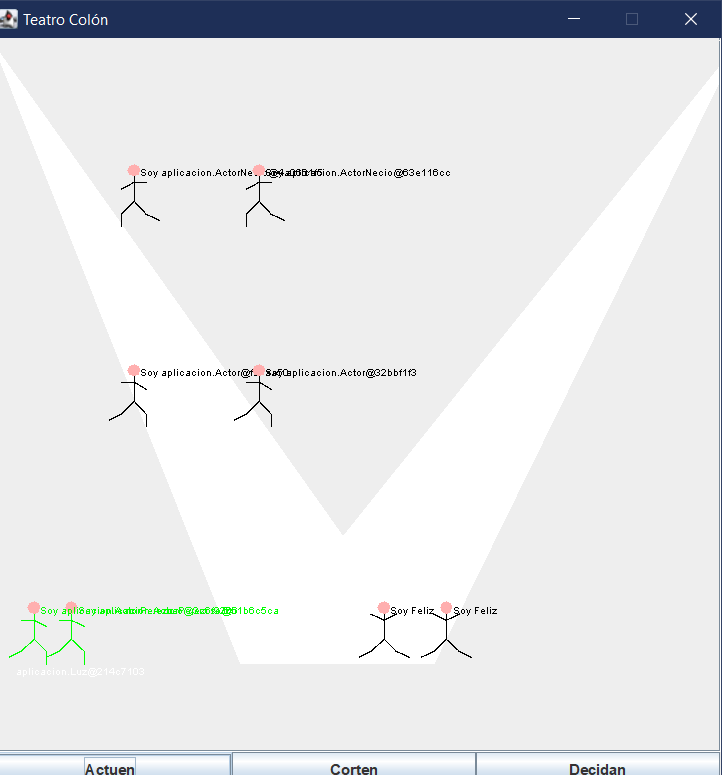
1. Incluyan una pareja de ellos con el nombre de ustedes. Ejecute el programa con doscasos significativos. Explique la intención de cada caso y capture las pantallas correspondientes.











# Ciclo 6. Nuevo elemento: Proponiendo y diseñando

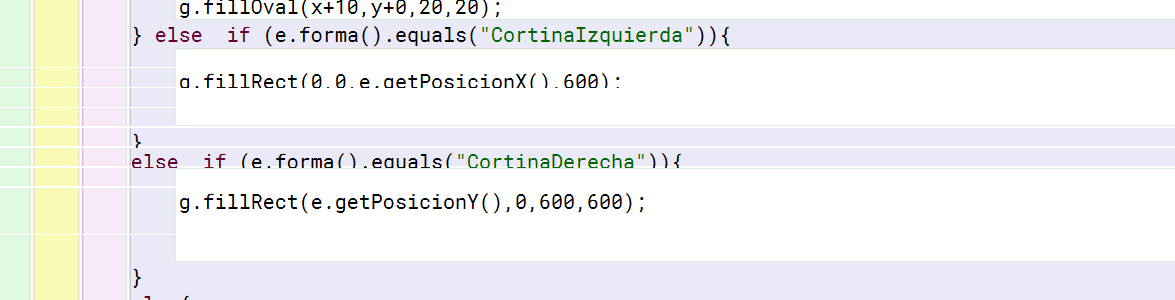
El objetivo de este punto es permitir recibir en un nuevo elemento en el escenario

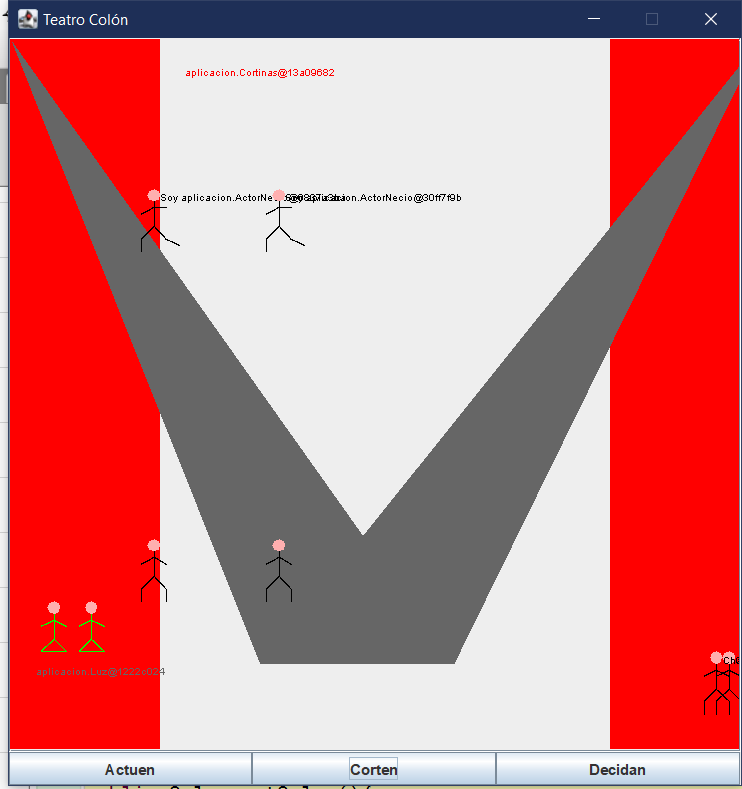
(NO OLVIDE BDD - MDD)

1. Propongan, describan e implementen un nuevo tipo de element

Creamos unas cortinas con rectangulos, para que hagan la function de cortinas que se cierran cuando se corta, cada corte se mueve 10 la cortina hacia adentro tanto de la derecho como la izquierda, cuando se juntan de nuevo se abren y empieza desde cero

1. Incluyan una pareja de ellos con el nombres semánticos. Ejecute el programa con dos casos significativos. Explique la intención de cada caso y capture las pantallas correspondientes.





**Empaquetando la versión final para el usuario.** [En lab03.doc, TeatroColon.asta , \*.java,

TeatroColon.jar]

1. Revise las opciones de BlueJ para empaquetar su programa entregable en un archivo .jar. Genere el archivo correspondiente.
2. Consulte el comando java para ejecutar un archivo jar. Ejecutelo ¿qué pasa?

java -jar NombreDelArchivoJAR

Al ejecutarlo se abre la Ventana de Teatro Colon

1. ¿Qué ventajas tiene esta forma de entregar los proyectos? Explique claramente.

Se tiene la interfaz gráfica directamente, y los demás datos están comprimidos dentro del archivo .jar, siendo esto una forma de comprimir todos los archivos en solo uno.

**DE BLUEJ A CONSOLA**

En esta sección del laboratorio vamos a aprender a usar java desde consola. Para esto se va a trabajar con el proyecto del punto anterior.

**Comandos básicos del sistema operativo** [En lab03.doc]

Antes de iniciar debemos repasar los comandos básicos del manejo de la consola.

1. Investiguen los comandos para moverse en la estructura de directorios: crear, borrar, listar su contenido y copiar o eliminar un archivo.

* Crear: COPY CON
* Borrar: del, erase
* Listar: dir
* Copiar: COPY archivo directorio
* Eliminar: del, erase

1. Organicen un nuevo directorio con la estructura propuesta para probar desde allí su habilidad con los comandos de consola. Consulten y capturen el contenido de su directorio teatroColon

src aplicacion presentacion

pruebas

1. En el directorio copien únicamente los archivos \*.java del paquete de aplicación . Consulte y capture el contenido de src/aplicacion

**Estructura de proyectos java** [En lab03.doc]

En java los proyectos se estructuran considerando tres directorios básicos.

automata src bin docs

1. Investiguen los archivos que deben quedar en cada una de esas carpetas y la organización interna de cada una de ellas.
2. ¿Qué archivos debería copiar del proyecto original al directorio bin? ¿Por qué?

Cópielos y consulte y capture el contenido del directorio que modificó.

**Comandos de java** [En lab03.doc]

1. Consulte para qué sirven cada uno de los siguientes comandos:

javac java javadoc jar

1. Cree una sesión de consola y consulte en línea las opciones de los comandos java y javac. Capture las pantallas.
2. Busque la opción que sirve para conocer la versión a que corresponden estos dos comandos. Documente el resultado.

**Compilando** [En lab03.doc]

1. Utilizando el comando javac, **desde el directorio raiz (desde automata con una sóla instrucción)**, compile el proyecto. ¿Qué instrucción completa tuvo que dar a la consola para compilar TODO el proyecto? Tenga presente que se pide un único comando y que los archivos compilados deben quedar en los directorios respectivos.
2. Revise de nuevo el contenido del directorio de trabajo y sus subdirectorios. ¿Cuáles nuevos archivos aparecen ahora y dónde se ubican?

**Documentando** [En lab03.doc]

1. Utilizando el comando javadoc, desde el directorio raiz, genere la documentación (API) en formato html, en este directorio. ¿cuál es el comando completo para generar esta documentación?
2. ¿Cuál archivo hay qué abrir para empezar a navegar por la documentación? Ábralo y capture la pantalla.

**Ejecutando**  [En lab03.doc]

1. Empleando el comando java, desde el directorio raiz, ejecute el programa. ¿Cómo utilizó este comando?

**Probando**  [En lab03.doc]

1. Adicione ahora los archivos del directorio pruebas y trate de compilar nuevamente el programa.Tenga en cuenta que estas clases requieren la librería junit 4.8. ¿Cómo se incluye un paquete para compilar? ¿Qué instrucción completa tuvo que dar a la consola para compilar?
2. Ejecute desde consola las pruebas . ¿Cómo utilizó este comando?. Puede ver

ejemplos de cómo ejecutar el“test runner”en: <http://junit.sourceforge.net/doc/cookbook/cookbook.htm>

1. Pegue en su documento el resultado de las pruebas

**Empaquetando** [En lab03.doc]

1. Consulte como utilizar desde consola el comando jar para empaquetar su programa entregable en un archivo .jar, que contenga los archivos bytecode necesarios (no las fuentes ni las clases de prueba), y que se pueda ejecutar al instalarlo en cualquier directorio, con solo tener la máquina virtual de java y su entorno de ejecución (JRE). ¿Cómo empaquetó jar ?
2. ¿Cómo se ejecuta el proyecto empaquetado?

**RETROSPECTIVA**

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

15 horas cada uno

1. **¿Cuál es el estado actual de laboratorio? ¿Por qué? (Para cada método incluya su estado)**

Terminado por la constancia, aunque fue complicado entener el concpeto de override con el cual tuvimos que hacer una investigacion para entender la manera correcta de usarlo

1. **Considerando las prácticas XP del laboratorio de hoy ¿por qué consideran que son importante?**

Porgramacion a pares e integracion Continua

1. **¿Cuál consideran fue su mayor logro? ¿Por qué? ¿Cuál consideran que fue su mayor problema? ¿Qué hicieron para resolverlo?**

Aprender a usar bien la consola para iniciar java entre otros

1. **¿Qué hicieron bien como equipo? ¿Qué se comprometen a hacer para mejorar los resultados?**

Aprender compartiendo los conocimientos mutuamente