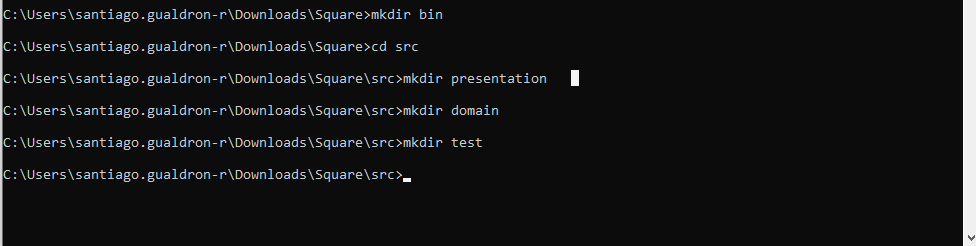
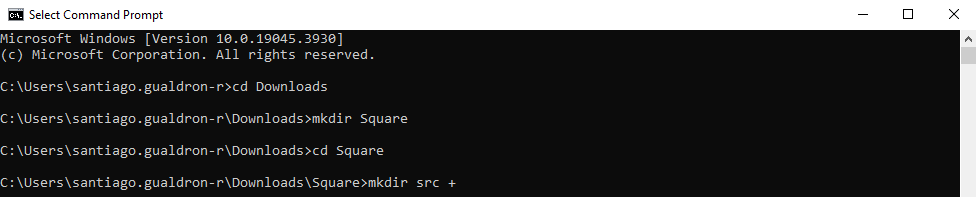
**Laboratorio 5**

**Santiago Gualdron –Sofia Gil**

**DESARROLLO**

**Directorios**

1. Preparen un directorio llamado Square con los directorios src y bin y los subdirectorios para presentación, dominio y pruebas de unidad.



**Ciclo 0: Ventana vacía – Salir [En \*.java y lab05.doc]**

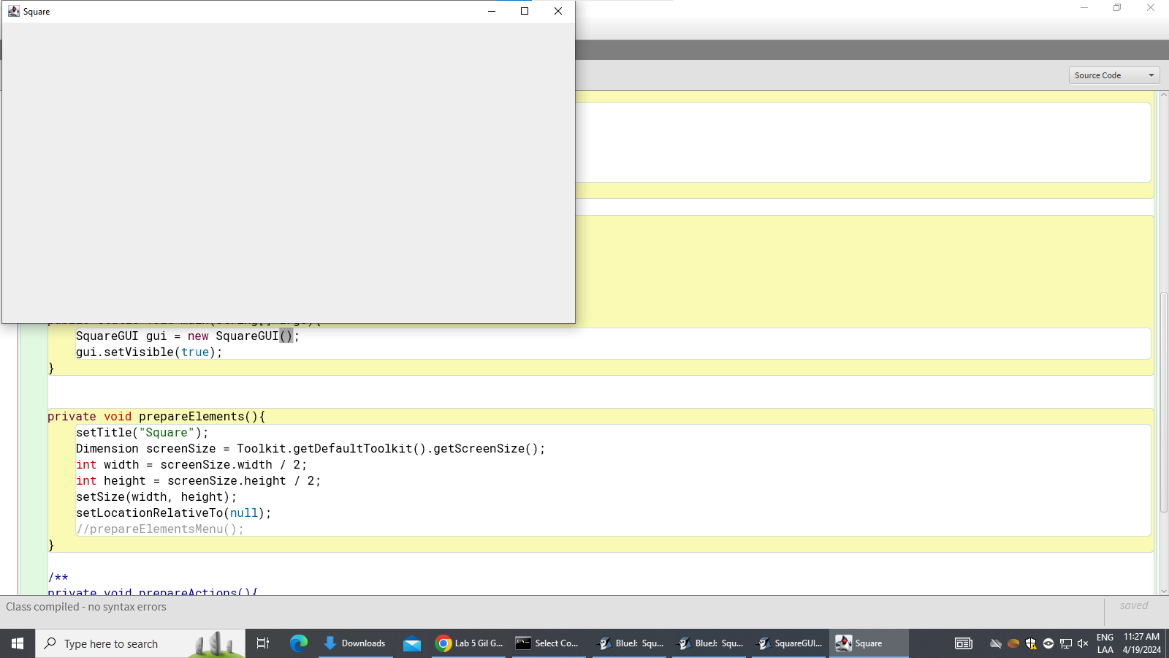
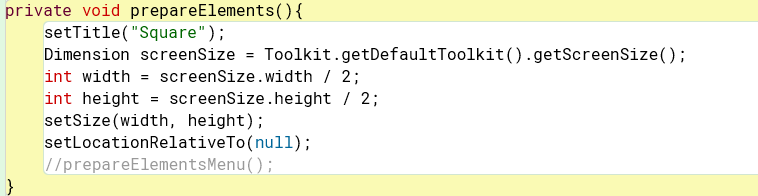
1. Construyan el primer esquema de la ventana de Square únicamente con el título “Square”. Para esto cree la clase SquareGUI como un JFrame con su creador (que sólo coloca el título) y el método main que crea un objeto SquareGUI y lo hace visible. Ejecútenlo. Capturen la pantalla. (Si la ventana principal no es la inicial en su diseño, después deberán mover el main al componente visual correspondiente)



Texto, Aplicación

Descripción generada automáticamente

2. Modifiquen el tamaño de la ventana para que ocupe un cuarto de la pantalla y ubíquenla en el centro. Para eso inicien la codificación del método prepareElements. Capturen esa pantalla.



Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

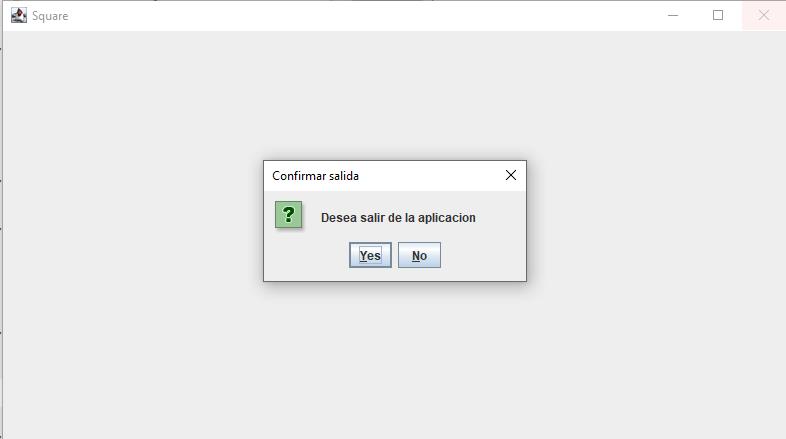
3. Traten de cerrar la ventana. ¿Termina la ejecución? ¿Qué deben hacer en consola para terminar la ejecución?

Para terminar la ejecución seleccionamos el botón de x, pero seguirá la aplicación ejecutándose en segundo plano, con System.exit(0) ya terminará toda la aplicación; en consola con Ctrl C ya detendrá la ejecución y seguirá el control de la consola.

4. Estudien en JFrame el método setDefaultCloseOperation. ¿Para qué sirve? ¿Cómo lo usarían si queremos confirmar el cierre de la aplicación? ¿Cómo lo usarían si queremos simplemente cerrar la aplicación?

En Java, setDefaultCloseOperation es un método de la clase Jframe que se utiliza para establecer la acción predeterminada que se realizara cuando el usuario intente cerrar la ventana del Jframe. Se podría usar teniendo en cuenta JFrame.DO\_NOTHING\_ON\_CLOSE: no realizar ninguna acción al intentar cerrar la ventana hasta que haya alguna confirmación.

5. Preparen el “oyente” correspondiente al icono cerrar que le pida al usuario que confirme su selección. Para eso inicien la codificación del método prepareActions y el método asociado a la acción (exit). Ejecuten el programa y cierren el programa. Capturen las pantallas.



**Ciclo 1: Ventana con menú – Salir [En \*.java y lab05.doc]**

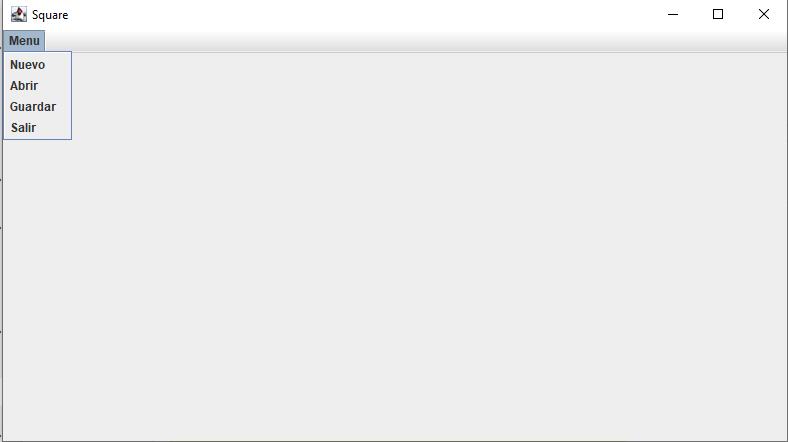
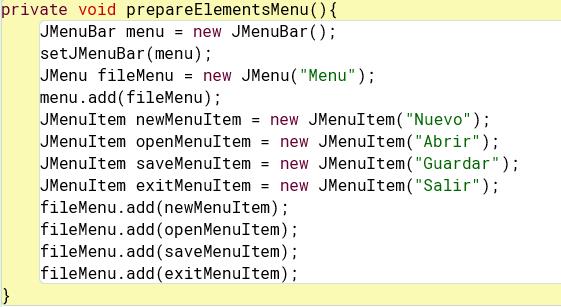
1. Expliquen los componentes visuales necesarios para este menú. ¿Cuáles serían atributos y cuáles podrían ser varibles del método prepareElements? Justifique.

JMenuBar: Este es el menú principal que contendrá todas las opciones del menú, como "Nuevo", "Abrir", "Guardar", "Salir", Este podría ser un atributo de la clase SquareGUI ya que representa un componente clave de la interfaz de usuario que permanece constante.

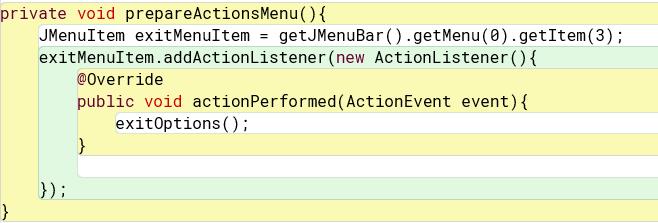
JMenu: Cada opción del menú, como "Nuevo", "Abrir", etc., es un elemento de menú (JMenuItem). Estos elementos de menú se agrupan bajo el menú principal (JMenu). En este caso, podrían ser variables dentro de prepareElements ya que son elementos específicos del menú que se configuran y agregan al menú principal.

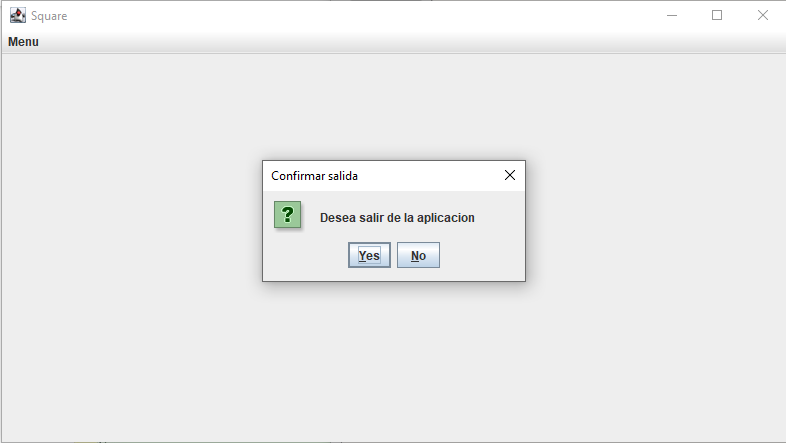
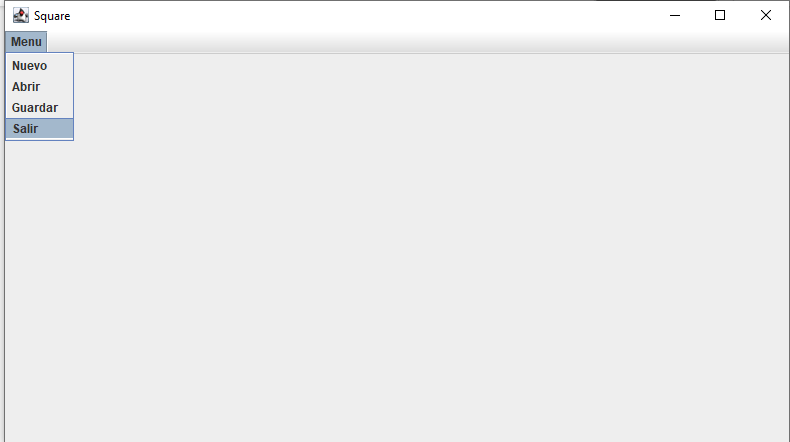
JMenuItem :Estos representan las opciones individuales dentro de un menú, lo que podrían ser variables de prepareElements ya que son elementos específicos del menú que se configuran y agregan al menú principal.

2. Construya la forma del menú propuesto (prepareElements - prepareElementsMenu) . Ejecuten. Capturen la pantalla.



3. Preparen el “oyente” correspondiente al icono cerrar con confirmación (prepareActions - prepareActionsMenu). Ejecuten el programa y salgan del programa. Capturen las pantallas.





**Ciclo 2: Salvar y abrir [En \*.java y lab05.doc]**

1. Detalle el componente JFileChooser especialmente los métodos : JFileChooser, showOpenDialog, showSaveDialog, getSelectedFile.

JFileChooser es un componente en Swing de Java que proporciona una interfaz gráfica para que los usuarios seleccionen archivos o directorios.

*1. showOpenDialog:*

* int showOpenDialog(Component parent)
* Muestra un cuadro de diálogo para abrir archivos.
* Devuelve JFileChooser.APPROVE\_OPTION si el usuario elige abrir, JFileChooser.CANCEL\_OPTION si cancela.

*2. showSaveDialog:*

* int showSaveDialog (Component parent)
* Muestra un cuadro de diálogo para guardar archivos.
* Devuelve JFileChooser.APPROVE\_OPTION si el usuario elige guardar, JFileChooser.CANCEL\_OPTION si cancela.

*3. getSelectedFile:*

* File getSelectedFile()
* Devuelve el archivo seleccionado después de que el usuario interactúa con el cuadro de diálogo.

2. Implementen parcialmente los elementos necesarios para salvar y abrir. Al seleccionar los archivos indique que las funcionalidades están en construcción detallando la acción y el nombre del archivo seleccionado.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente



3. Ejecuten las dos opciones y capturen las pantallas más significativas.

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamenteInterfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente

**Ciclo 3: Forma de la ventana principal [En \*.java y lab05.doc]**

1. Presenten el bosquejo del diseño de interfaz con todos los componentes necesarios.

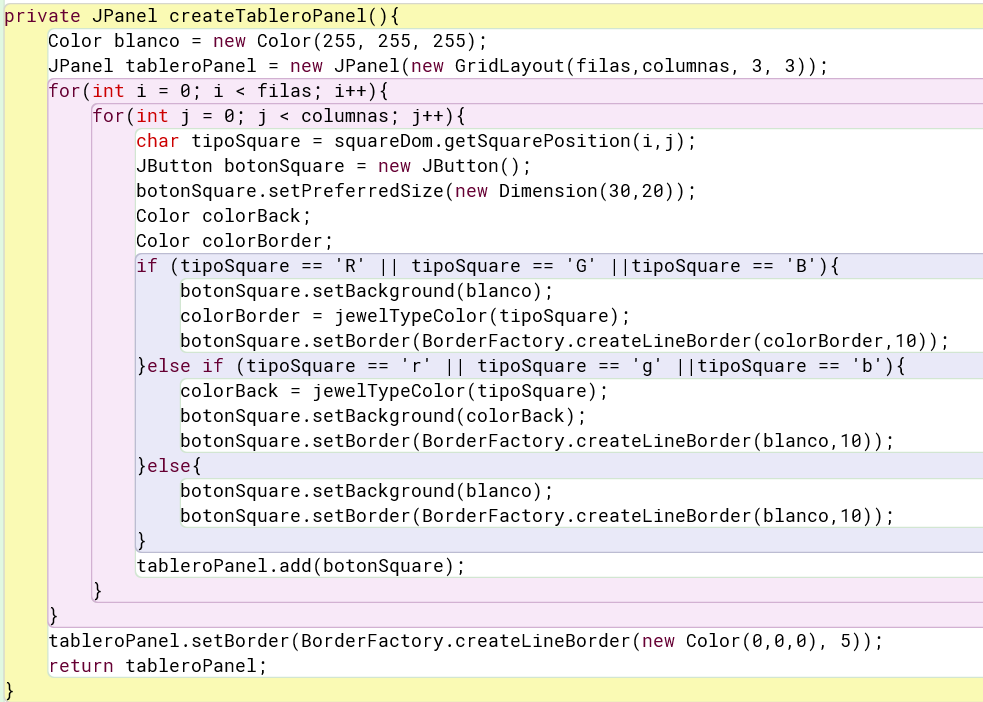
Gráfico de barras

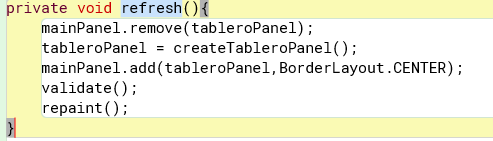
Descripción generada automáticamente con confianza baja

2. Continúe con la implementación definiendo los atributos necesarios y extendiendo el método prepareElements(). Para la zona del tablero defina un método prepareElementsBoard() y un método refresh() que actualiza la vista del tablero considerando, por ahora, el tablero inicial por omisión. Este método lo vamos a implementar realmente en otros ciclos.

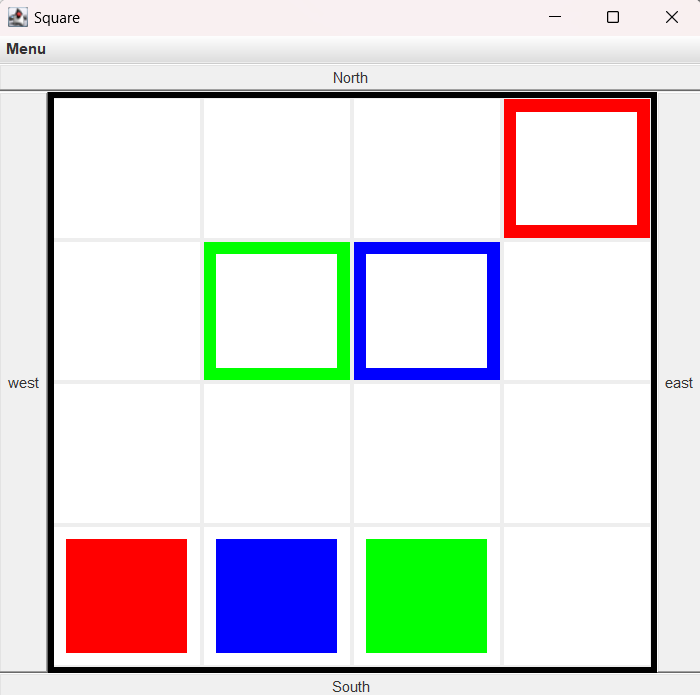
Texto

Descripción generada automáticamente





3. Ejecuten y capturen esta pantalla.



**Ciclo 4: Cambiar color [En \*.java y lab05.doc]**

1. Expliquen los elementos (vista – controlador) necesarios para implementar este caso de uso.

Elementos de la Vista :

Componentes de Selección de Color: Se necesitará componentes que permita al usuario seleccionar un color, como un diálogo de selección de color (JColorChooser) que se puede abrir al hacer clic en un botón Cambiar Color.

Botón Cambiar Color: Este botón es un componente de GUI que el usuario puede hacer clic para iniciar el proceso de cambio de color.

Elementos del Controlador:

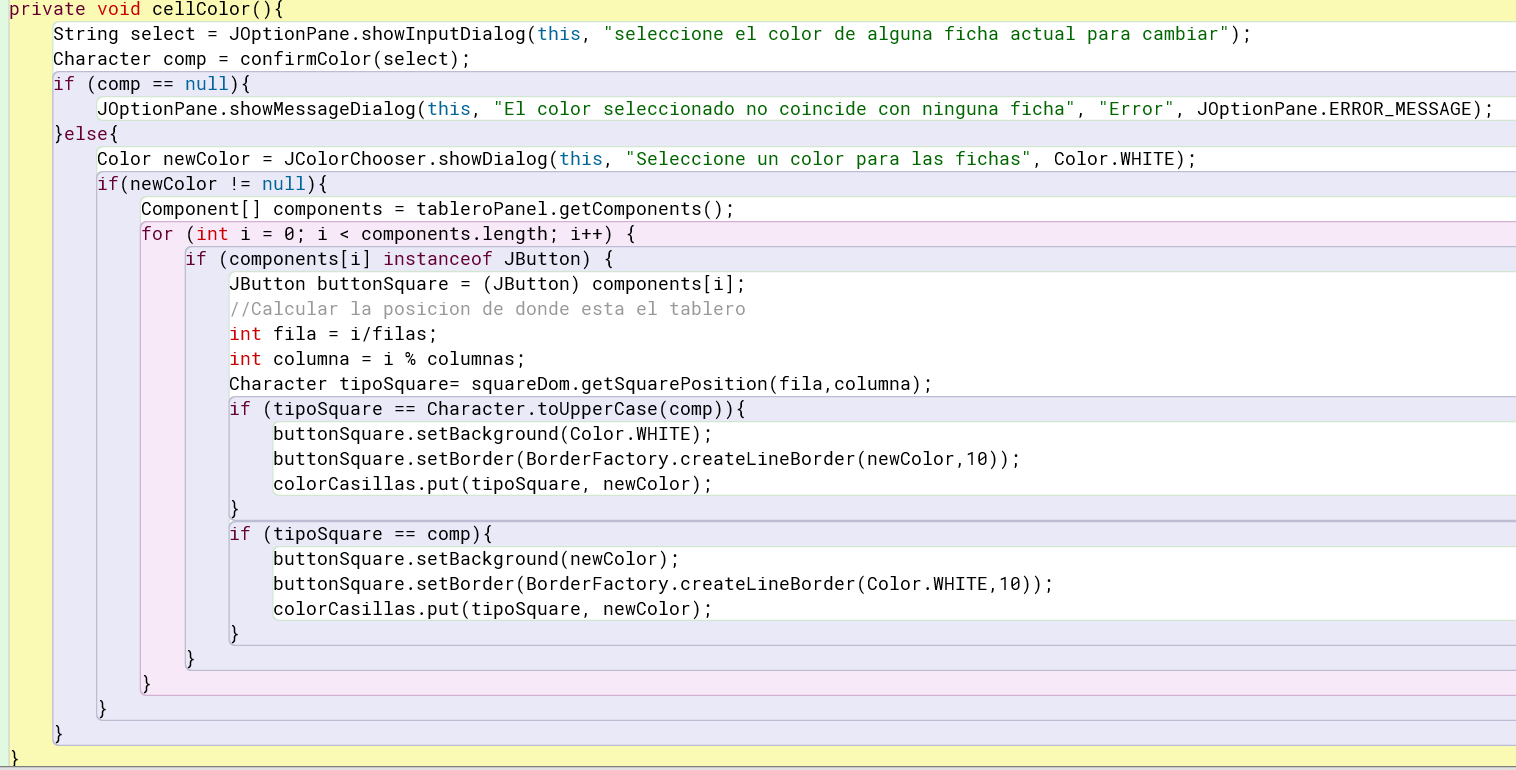
Evento de Cambio de Color: Se debe asociar un evento al botón Cambiar color para que cuando el usuario haga clic en él, se inicie el proceso de cambio de color.

Manejo del Evento de Cambio de Color: El controlador debe tener un método que maneje el evento de cambio de color.

2. Detalle el comportamiento de JColorChooser especialmente el método estático showDialog

JColorChooser es una clase en Java que proporciona una interfaz gráfica para que los usuarios seleccionen un color. El método estático showDialog se utiliza para mostrar el cuadro de diálogo de selección de color.

3. Implementen los componentes necesarios para cambiar el color de las fichas.



4. Ejecuten el caso de uso y capture las pantallas más significativas.

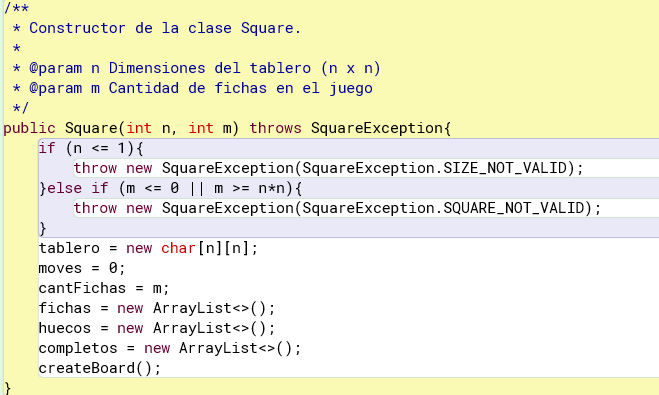
Interfaz de usuario gráfica

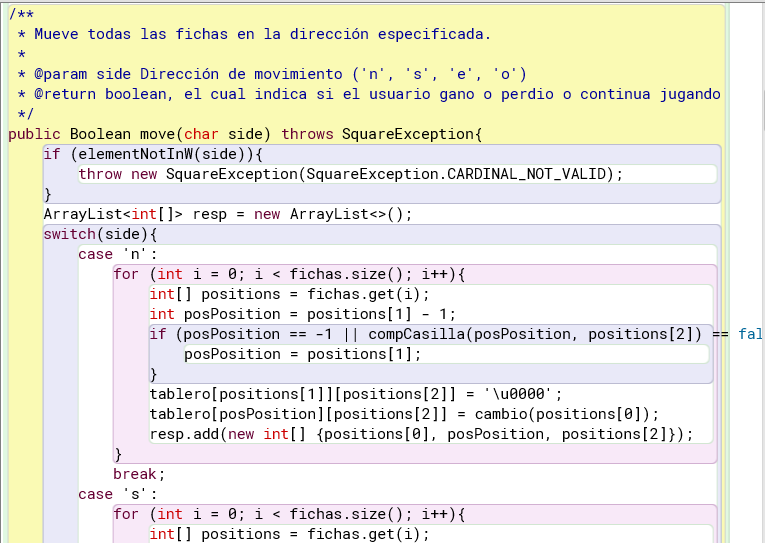
Descripción generada automáticamente Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

**Ciclo 5: Modelo Square [En \*.java y lab05.doc]**

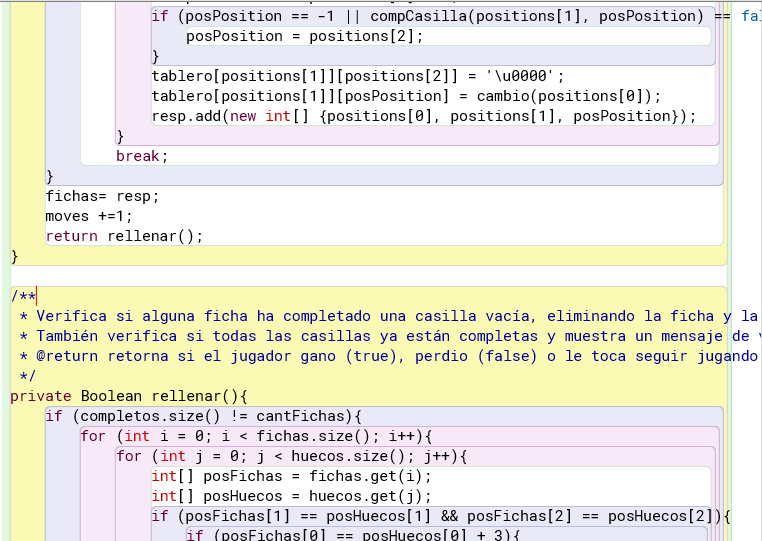
1. Construya los métodos básicos del juego (No olvide MDD y TDD)

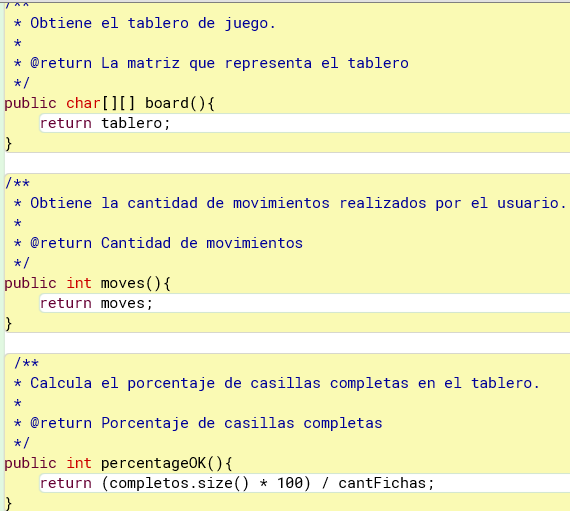




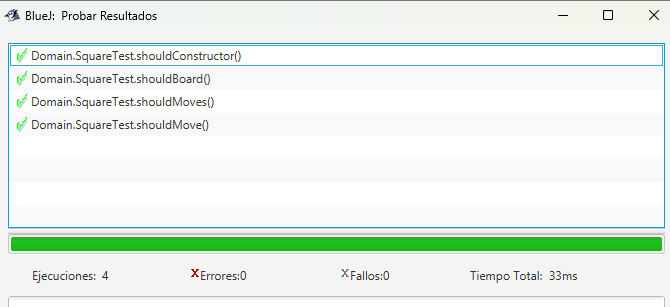
Texto

Descripción generada automáticamente



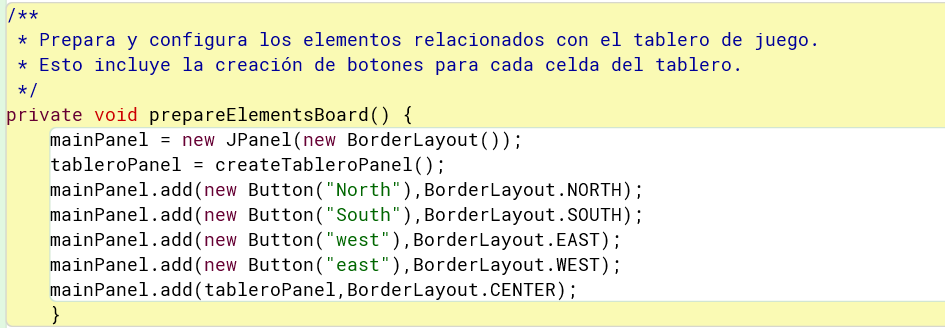


2. Ejecuten las pruebas y capturen el resultado.



**Ciclo 6: Jugar [En \*.java y lab05.doc]**

1. Adicione a la capa de presentación el atributo correspondiente al modelo.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

2. Perfeccionen el método refresh() considerando la información del modelo de dominio.

Para implementar el refresh, una vez que se hace el dominio y el método move; presentation debe conocer cual es la nueva matriz, por lo que el método refresh debe buscar la nueva matriz y repintarla.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

3. Expliquen los elementos necesarios para implementar este caso de uso.

Los elementos necesarios son el dominio específicamente con el método move(), 4 botones cardinales que ayuden a indicar hacia donde va las fichas, y un método mas que valla a estar atento si el jugador gana, pierde o puede seguir jugando.

4. Implementen los componentes necesarios para jugar .¿Cuántos oyentes necesitan? ¿Por qué?

Se necesitan 4 oyentes, los cuales son para los botones: “North”, “South”, “East”, “West”.

Además de tener un oyente que revise cuando el jugador gano o perdió, y preguntarle que desea hacer a continuación.

5. Ejecuten el caso de uso y capture las pantallas más significativas.

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamenteImagen que contiene Escala de tiempo

Descripción generada automáticamenteInterfaz de usuario gráfica

Descripción generada automáticamente

**Ciclo 7: Reiniciar [En \*.java y lab05.doc]**

1. Expliquen los elementos a usar para implementar este caso de uso.

Elementos de la Vista:

Botón Reiniciar: Este botón es un componente de GUI que el usuario puede hacer clic para iniciar el proceso de reinicio del juego.

Mensaje de éxito: Se debe mostrar un mensaje de éxito al usuario cuando se haya reiniciado el juego.

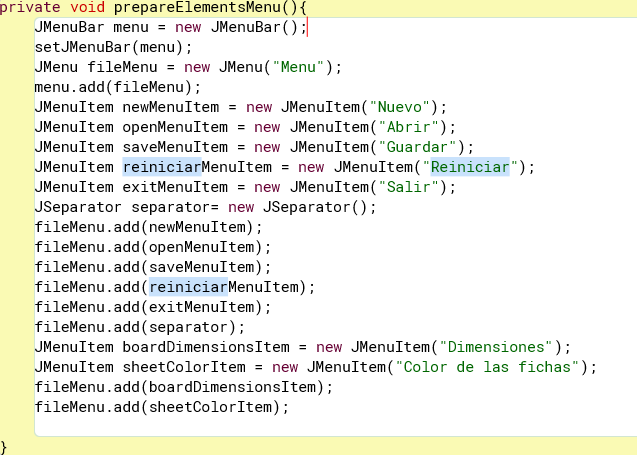
Elementos del Controlador:

Evento de Reinicio: Se debe asociar un evento al botón Reiniciar para que cuando el usuario haga clic en él, se inicie el proceso de reinicio del juego.

Manejo del Evento de Reinicio: El controlador debe tener un método que maneje el evento de reinicio.

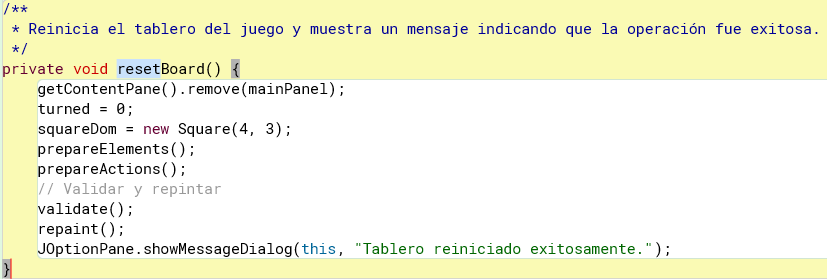
Lógica de Reinicio: El controlador debe contener la lógica necesaria para reiniciar el estado del juego a su estado inicial.

2. Implementen los elementos necesarios para reiniciar



Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

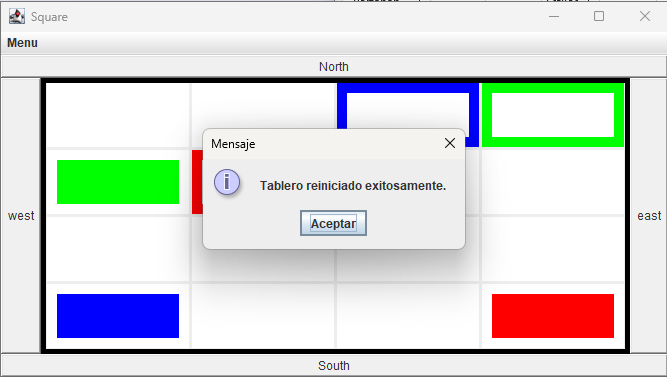
Descripción generada automáticamente



3. Ejecuten el caso de uso y capture las pantallas más significativas.

Gráfico

Descripción generada automáticamente con confianza media



**Ciclo 8: Cambiar el tamaño [En \*.java y lab05.doc]**

1. Expliquen los elementos a usar para implementar este caso de uso

Elementos de la Vista:

Botón Dimensiones: Este botón es un componente de GUI que el usuario puede hacer clic para iniciar el proceso de cambiar el tamaño del juego.

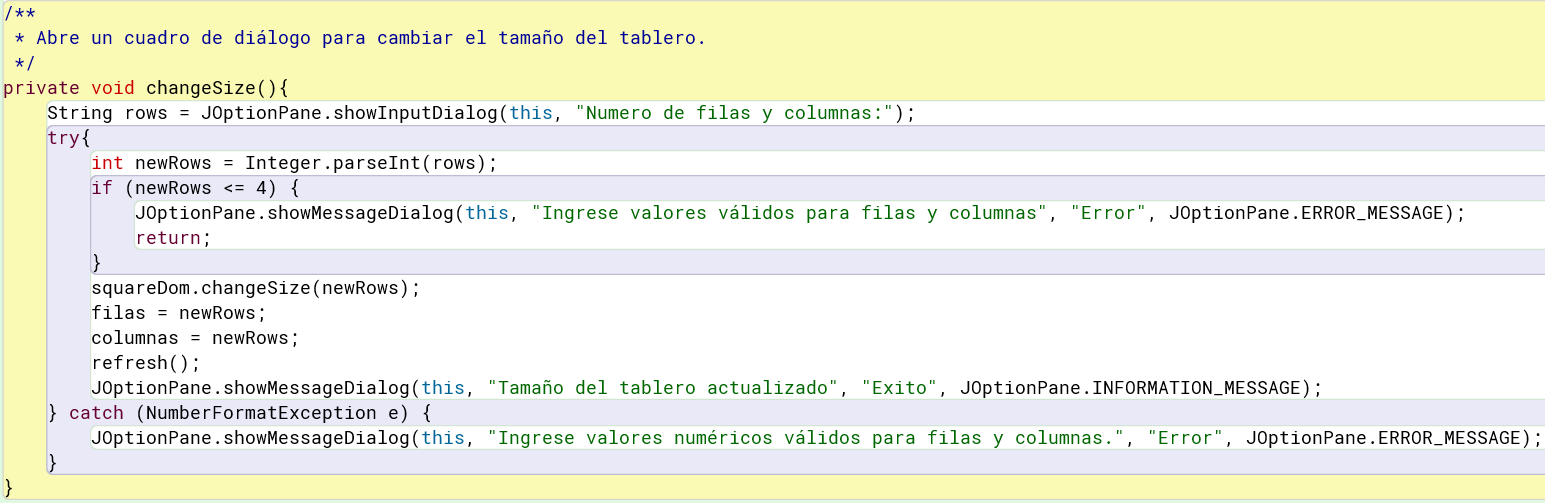
Diálogo de Configuración de Tamaño: Se debe mostrar un diálogo donde el usuario pueda ingresar las nuevas dimensiones (número de filas y columnas) para cambiar el tamaño del tablero.

Elementos del Controlador:

Evento de Cambio de Tamaño: Se debe asociar un evento al botón Dimensiones para que cuando el usuario haga clic en él, se abra el diálogo de configuración de tamaño.

Manejo del Evento de Cambio de Tamaño: El controlador debe tener un método que maneje el evento de cambio de tamaño.

2. Implementen los elementos necesarios para cambiar el tamaño del juego



3. Ejecuten el caso de uso y capture las pantallas más significativas.

Interfaz de usuario gráfica

Descripción generada automáticamenteInterfaz de usuario gráfica, Aplicación, Tabla, Excel

Descripción generada automáticamente

**RETROSPECTIVA**

1. ¿Cuál fue el tiempo total invertido en el laboratorio por cada uno de ustedes? (Horas/Hombre)

20 horas en total (10 por persona)

2. ¿Cuál es el estado actual del laboratorio? ¿Por qué?

El laboratorio quedo completo, en este caso conseguimos mejorar el tema de la optimización del tiempo y el comprendimiento de los elementos que pedía el laboratorio.

3. Considerando la práctica XP del laboratorio ¿por qué consideran que es importante?

Utilizamos la practica XP pair programming, la cual sirve bastante para saber como se estaba creando el paquete de presentación y el GUI; además de la refactorización de código donde varias veces nos toco reformular partes del código que sobraban o daban incoherencias

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

El mayor logro fue obtener el GUI de manera completa, ya que esta zona no tenia una base la cual pudiéramos basarnos, además de tener que pensar en como se tendría que hacer el diseño del juego a nuestro modo.

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

Hicimos bien el tema de la comprensión, además de estar constantemente contándole al otro que ha sido lo que se adelanta de forma individual.

Nos comprometemos a intentar poner en práctica más practicas XP.