TRABAJO PRÁCTICO

PROGRAMACIÓN III

ROMPECABEZAS DESLIZANTE

Alumnos: Aranda Bao Ignacio, Melhado Dante y Suarez Emanuel.

Profesores : Bagnes Patricia y Sotelo Ignacio.

2do Cuatrimestre 2024

INTRODUCCIÓN

En el presente trabajo práctico se plantea una implementación del clásico juego de Rompecabezas Deslizante, dado como consigna para el primer trabajo práctico de la materia. En este trabajo se utilizarán todos los conceptos aprendidos durante la cursada con el fin de presentar un buen proyecto en el cuál están reflejados los temas dados por la materia.

Se va a tomar muy en cuenta el concepto de "Separated presentation", en el cuál el código que se utiliza en la interfaz debe estar bien encapsulado y separado del código de negocio que es el encargado de manejar toda la lógica del juego. Se utilizará la arquitectura de GUI "Forms and Controls", la cual permite cumplir el principio de "Separated presentation".

Nuestro diseño busca lograr la mayor cohesión entre los métodos de las clases buscando que estén relacionados entre sí, delegando las responsabilidades entre las clases, y que haya el menor acoplamiento posible teniendo en cuenta que cada clase tenga sus responsabilidades bien definidas, a fin de poderse leer, interpretar y poder readaptar el código a diferentes usos.

Con el patrón de diseño "Forms and Controls" tendremos interfaces visuales (llamadas frames) con controles visuales, y las interfaces realizan llamados al código de negocio. Este código de interfaz realiza llamados al código de negocio, para ejecutar acciones solicitadas por el usuario (En este caso, mover el casillero vacío) y actualizar los controles visuales cuando cambia el estado del sistema (Al mover el casillero vacío, cambia de lugar con otros casilleros).

Este proyecto se separó en 3 paquetes, un paquete con el main que inicializará el juego, un paquete llamado "interfaz" en el cual tendrá todas las clases relacionadas a la interfaz visual del programa que sería la interfaz visual. Y otro paquete llamado "lógica" en el cual se tendrá todo el código de negocio que respecta a la lógica del juego en sí, como movimientos, puntuaciones, etc.

PRESENTACIÓN - REGLAS DEL JUEGO

Al iniciar la ventana aparecerán las ventanas de Iniciar, Opciones y Salir.

Al interactuar con Opciones, se podrá elegir tanto el tamaño del tablero (3x3, 4x4, 5x5, 6x6 y 7x7) como la dificultad del mismo (Fácil, Normal y Difícil). Dicha dificultad lo que hará será agregar mayor aleatoriedad al tablero inicial.

Luego al iniciar el juego, el jugador podrá desplazarse con las teclas de dirección (`↑`, `←`, `↓`, `→`) o las letras `W`, `A`, `S`, `D` para mover el espacio en blanco con el objetivo de ordenar los números del tablero y así ganar el juego. El sistema de puntaje es del mismo estilo que el Golf, en donde cuantos menos golpes (movimientos en este caso) mejor.

Por último, dentro de esta pantalla se encuentran dos pestañas arriba a la izquierda (Juego y Ayuda). Con Juego se puede regresar al menú principal o salir del juego. Con Ayuda se accede a Acerca de (autores del programa) y Cómo jugar (reglas).

IMPLEMENTACIÓN

Al principio de la implementación se pensó con qué arquitectura de GUI se iba a trabajar. Al decidir que se implementaría en torno a la arquitectura "Forms and Controls" se pensó en separar las interfaces visuales del código de negocio o la lógica en sí. Se separó en dos paquetes en la cual uno se encargaría de las interfaces visuales y el otro se va a encargar todo en lo que respecta al código de negocio.

En la generación de tableros aleatorios se utiliza la estrategia recomendada por los docentes: empezar con el tablero ya resuelto y luego aplicar una serie de movimientos al azar. Se hace de esta manera puesto que si simplemente se generan los números del tablero se puede dar la posible aparición de paridades, que son imposibles de resolver y verificar que un tablero no tenga tal paridad es muy complejo.

Se explicaran a continuación brevemente las clases dentro de cada paquete:

Paquete Interfaz - Interface Visual.

En el paquete interfaz vamos a tener todas las clases que se van a encargar de manejar las interfaces visuales (frames). Acá tendremos múltiples clases de las cuales explicaremos qué hace cada una:

- **clase MainFrame**: esta clase es la encargada de crear el JFrame en el cual vamos a interactuar durante todo el juego y de manejar entre los cambios de paneles. La clase va a trabajar con CardLayout, que lo que nos va a permitir este tipo de diseño es que vamos a poder mostrar diferentes paneles en una misma ventana.
- **clase VentanaMenu**: JPanel que nos va a recibir ni bien iniciaremos el proyecto en la cual va a tener un JPanel en la cual tendremos JButtons que nos dirigirá a la ventana opciones, como a la ventana del juego en sí y la opción de salir del juego.
- **clase VentanaJuego**: jPanel encargado de mostrar en pantalla el juego en sí y mostrarnos en todo momento el progreso del juego, en tanto a puntuaciones y los movimientos que realiza el jugador.
- **clase VentanaOpciones**: JPanel que se ocupa de que el usuario pueda elegir la dificultad y el tamaño que quiera.
- **clase VentanaReglas**: este es un JDialog en el cual va a explicar las reglas del juego para quien no sepa como jugar.
- **clase Fuente**: clase que se va a encargar de cargar una fuente personalizada desde un archivo externo, de la cual va a devolver un Font que se va a utilizar en la interfaz gráfica.

Paquete Lógica - Lógica de Negocio.

En el paquete lógica vamos a tener todas las clases que se van a encargar de implementar el código de negocio del programa.

- **clase GameBoard**: esta sería la clase principal de la lógica del juego, ya que va a manejar casi toda la lógica del movimiento y los estados del juego.
- **clase Score**: clase encargada de todo el sistema de puntuación en cuanto a los movimientos del jugador.
- **clase Direction:** esta clase es una enumeración que define las 4 constantes de la cual se utilizarán para representar los movimientos o direcciones de teclado.

PROBLEMAS ENCONTRADOS

1- WindowBuilder

El mayor desafío que tuvimos a la hora de realizar el trabajo fue la de comenzar a utilizar el WindowBuilder. Al ser el primer acercamiento con esta herramienta, hubo muchos problemas a la hora de colocar los botones, darles tamaño y modificar sus fuentes. Además inicialmente habíamos colocado todas las interfaces juntas, y tuvimos que separarlas más adelante.

2- Gestión de Paneles

Se tuvo un gran problema en lo que respecta a gestión de ventanas, ya que al principio la clase MainFrame manejaba las 3 ventanas principales (VentanaJuego, ventanaOpciones, ventanaMenu), el programa funcionaba pero la clase MainFrame era muy extensa ya que tendría que manejar tanto la ventana de juego como la de opciones y la del menú en sí. Por lo tanto, se pensó separar las ventanas en 4 clases (MainFrame, VentanaJuego, VentanaOpciones, VentanaMenu). Al intentar separar las ventanas en clases diferentes, se tuvo problemas ya que no se podían cambiar entre las ventanas, o se iniciaba en una pero nunca se lograba entrar a jugar. Luego de un arduo trabajo de buscar soluciones, se encontró que la solución estaba en usar CardLayout. CardLayout es un administrador de diseño que vamos a poder utilizar para manejar múltiples paneles en un contenedor (JFrame), que permite que solo uno esté visible a la vez. Esto nos permitió alternar entre los diferentes paneles, y la correcta gestión de los mismos.

3- Colocación de Imagen en Juego.

Se quiso colocar imágenes en lugar de números en el tablero, y pudimos implementarlo pero no encontramos forma de ajustar la imagen cuando se agrandaba la ventana del designer, por lo cual desistimos de esa idea para enfocarnos en otras tareas.

4- Bug de Tablero

Nos encontramos con un problema al iniciar el juego debido a que si se hacía click en cualquier botón de la matriz el programa dejaba de leer las teclas. La solución fue aplicar un

setFocusable(false) a cada uno de los botones luego de que el juego comience y se apliquen los randoms.