UNIVERSIDAD MAYOR DE SAN SIMÓN

FACULTAD DE CIENCIAS Y TECNOLOGÍA

Taller de Programación

Proyecto Grupal

Documentación Final

del Juego “Slide Puzzle”

Docente: Víctor Hugo Montaño Quiroga **Carrera:** Ingeniería de Sistemas

Integrantes:

Ezequiel Gerstel Bodoha

Adan Alberto Llanos Zela

Christian Mauricio Arias Chubarieva

Semestre II/2021

Cochabamba – Bolivia

Índice

Contenido

1. Introducción

Una parte importante del mercado digital de hoy está relacionado a los videojuegos y a los recursos de ocio proporcionados a los consumidores en sus dispositivos. Dentro de este grupo, una de las primeras implementaciones de este tipo era llevar juegos clásicos a una versión digitalizada, ya sean juegos de cartas como el póker o el solitario, o dinámicas más casuales como el conocido “tres en raya”. Dentro de este panorama se presenta un juego lógico simple pero llamativo, que consisten en ordenar un rompecabezas con unas reglas muy particulares.

2. Descripción detallada del problema

El problema consiste en desarrollar un juego funcional, una versión digital de los conocidos “slide puzles” o puzzles deslizantes. En esta se debe incluir el juego clásico, una rejilla de 3x3 con una imagen desordenada que deberá ser resuelta moviendo las fichas de lugar de manera vertical y horizontal hasta llegar a la imagen ordenada.

Se incluyen ampliaciones digitales, que incluyen el uso de un cronómetro, un registro de las partidas de otros jugadores y sus resultados, además de la posibilidad de recordar en qué estado se dejó un juego y recuperarlo más tarde.

Pero además de esto, se desea ofrecer una amplia posibilidad de personalización, difiriendo así mucho del juego clásico y llevándolo a un nivel mucho más competitivo. La rejilla no es estricta en 3x3, ni limitada al mismo número de filas y columnas. Asimismo, la imagen empleada en el juego puede ser elegida entre las opciones de su preferencia, o en última instancia, ser una seleccionada por el propio jugador para darle su propio estilo.

Se deben cumplir con ciertos estándares y canones generales de este tipo de aplicaciones, que incluyen funcionalidades normales e intuitivas para una buena navegación por parte del usuario.

3. Objetivos

3.1.General

Desarrollar un juego funcional y de uso general. Funcional porque es capaz de adaptarse a las necesidades del usuario sin que su funcionamiento se modifique o detenga. Y general porque debe ser suficientemente simple para ser usado por cualquier persona, y pueda centrarse en el juego olvidando que está usando una aplicación de computador.

3.2.Específicos

Brindar información relevante al usuario tal como el tiempo de juego, los pasos realizados, los restantes, y pausar y continuar el juego a gusto.

La interfaz debe ser clara y con pocos elementos debe proporcionar toda la información relevante para una navegación cómoda y efectiva.

Proveer de herramientas que mejoren la experiencia de juego, como la posibilidad de guardar/cargar partida, tabla de puntajes entre otros.

Ser estéticamente decente. Lógicamente no es un juego de última generación, pero tampoco debe verse como una página antigua en HTML, debe poseer cierto diseño agradable y que incite a jugar.

4. Requerimientos de Usuario

El usuario debe lograr jugar de forma libre e intuitiva, siendo capaz de acceder a una ventana de configuración previa al juego, donde preparar el juego a su gusto y añadir información que será guardada posteriormente. Se le debe proveer de herramientas variadas para que la aplicación no se sienta monótona.

Entre estos requerimientos se debe encontrar la posibilidad de configurar la dificultad del puzzle a gusto con algunas restricciones menores (debido a la naturaleza del juego), dando pie a no solo un juego cuadrado, sino más bien variado y complejo si es necesario. Asimismo, se requiere que pueda elegir una imagen personalizada de cualquier directorio, y poder jugar con el cuadro que el desee.

Otro punto importante es la gestión de la partida, tanto su guardado y cargado deben estar debidamente implementados, y siendo que los datos de juego se

almacenen y mantengan intactos hasta su propia activación. El usuario debe confiar en que el programa será de preguntarle apropiadamente cuando debería guardar la partida, previniendo así inconvenientes y situaciones no deseadas.

Por último, a lo largo de todas las etapas se deben cumplir algunos requerimientos estándar, como una ventana de ayuda que provea al usuario de las instrucciones básicas para jugar, de la posibilidad de observar los mejores puntajes, y también abandonar el juego de manera cómoda. Requiere por supuesto, que la aplicación funcione debidamente en todo momento y tenga una barrera que produzca errores, por más que el usuario realice acciones poco comunes.

5. Requerimientos Funcionales

Una de las primeras dificultades presentadas va en relación con el uso de una interfaz amigable, no solo para el usuario, sino para los desarrolladores. Es necesario el uso de una herramienta práctica y cómoda para editar la parte gráfica del juego, dado que la naturaleza del mismo depende altamente en los componentes visuales. Debe ser además de uso liviano, que funcione apropiadamente en cualquier dispositivo moderno, con tiempos de carga muy reducidos o nulos, representando una respuesta casi inmediata a cada proceso.

Otro requerimiento es la comunicación entre la parte lógica del programa y su componente de interfaz. Existen muchos procesos vitales, que, de presentarse en el área gráfica, resultarían incómodos o inapropiados para el usuario (como la solución visible del puzzle), y por tanto deben ser gestionados por debajo por el programa. Pero al mismo tiempo, la información procesada por la parte lógica del programa debe ser de alguna forma leída y recuperada por la interfaz que el usuario verá, para representar en tiempo real una interacción funcional.

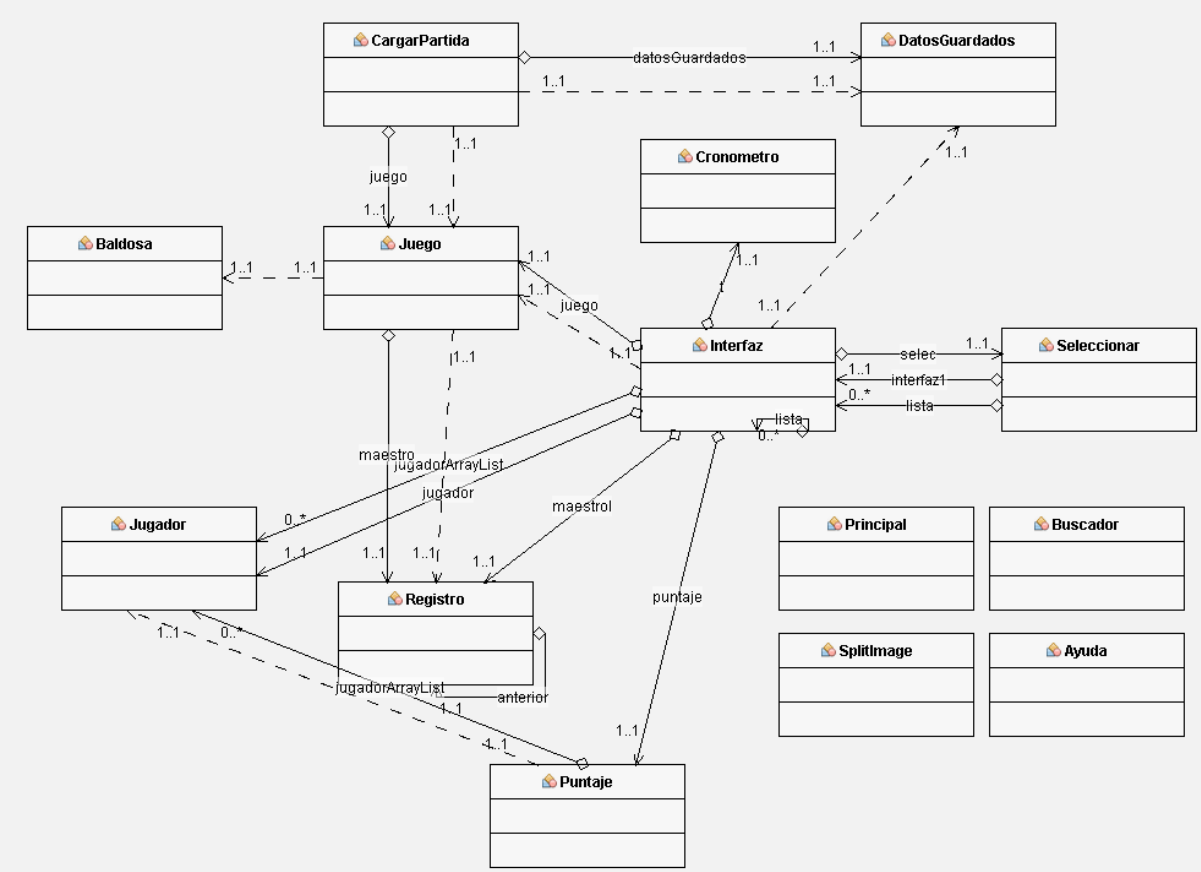
Dado que se provee al usuario de una gran libertad al usar la aplicación, la misma debe responder apropiadamente ante sus necesidades, no presentado huecos y solucionando los posibles escenarios planteados siempre de una forma limpia y elegante. Para ello se proponen soluciones generales que no solo resuelvan un problema, sino una serie de posibles problemas, dando pie a un código robusto.

6. Análisis (modelamiento)

6.1.Modelo de Clases

A continuación, se muestra el diagrama de clases de la aplicación.

Diagrama de Clases UML



7. Especificación de Clases

7.1.Juego

Es la lógica del juego, se encarga de mover las baldosas una vez realizado un movimiento utiliza los métodos de la clase “Registro” para almacenarlo. Implementa las opciones de desordenar y ordenar con una solución general el puzzle.

Se ordena el puzzle haciendo uso de los registros de movimientos, al desordenar se registran todos lo movimientos aleatorios que el método realiza de igual forma se registran los movimientos cuando el jugador mueve las baldosas. El método “ordenar()” es el que deshace todos los movimientos realizados, de esta forma soluciona el puzzle.

Para ordenar el puzzle se deshacen los movimientos almacenados en la “pila” de datos de la clase “Puntaje”; si la “pila” detecta que se deshacen movimientos, automáticamente se borra el registro del movimiento dando lugar al siguiente, se hace este proceso hasta vaciar los registros.

7.2.Interfaz

Es la interfaz gráfica del juego, contiene elementos visuales con los que puede interactuar el usuario, dentro de esta clase están diseñados los indicadores, el tablero del puzzle, el menú de opciones y los botones.

Es una clase importante porque utiliza las demás clases que son parte de la lógica del juego y las implementa con elementos visuales.

7.3.Seleccionar

La clase seleccionar es la que contiene las opciones de configuración para crear una partida, su interfaz está compuesta por los paneles “ventanaInicio” y el panel “ventanaSelec”. La “ventanaInicio” es la presentación de la aplicación y la “ventanaSelec” ofrece opciones de configuración al usuario para crear una partida.

Dentro de las opciones de configuración, el usuario puede introducir su nombre, elegir la dificultad del puzzle y una imagen, ya sea predeterminada o una personalizada.

7.4.Baldosa

Esta clase crea la lógica de las baldosas del puzzle, funciona en conjunto con la clase “Juego” y se encarga de la construcción del tablero del puzzle. Dentro del constructor, cada baldosa se crea recibiendo su posición y su condición de si está vacía o no.

7.5.Registro

Es una clase importante porque es la que realiza los registros de los movimientos de las baldosas, con estos registros el programa puede resolver el puzzle sin importar el tamaño dando lugar a una solución general.

Los registros de movimientos son almacenados en una “pila”, que es una estructura de datos.

7.6.Puntaje

La interfaz de esta clase guarda la información como el nombre, el tiempo y los movimientos realizados de todos los jugadores que han completado el puzzle. Posteriormente realiza una operación para determinar los puntajes basados en esta información y se muestra en una tabla.

La tabla muestra en orden primero a los mejores jugadores, que son los que completaron el puzzle en la menor cantidad de movimientos y en el menor tiempo posible. Se puede hacer una búsqueda de los jugadores por nombre y por posición en el “ranking” en los cuadros de texto.

7.7.Principal

En esta clase está el método “main”, que ejecuta toda la aplicación.

7.8.Split Image

Es la clase que se encarga de recortar la imagen recibiendo por parámetros la ruta relativa de la imagen, la cantidad de filas y columnas en las que se dividir la imagen. Al momento de recortarlas se almacenan estas imágenes recortadas en nuestro dispositivo en la carpeta “src/images/galería” con un nombre “img” y con un numero especifico, si ya existe una imagen con el mismo nombre este se sobrescribirá.

7.9.Ayuda

Proporciona información al usuario, una breve explicación para empezar a jugar.

7.10. Buscador

Es la herramienta que permite abrir el directorio de archivos locales del usuario al momento de guardar y cargar partida.

7.11. CargarPartida

Esta clase implementa la clase “Serializable”, recibe como parámetros la clase juego y la clase “DatosGuardados” para poder hacer el guardado de partida en un archivo “.txt” para luego posteriormente recuperarlo y cargar la partida.

7.12. DatosGuardados

Esta clase solo se encarga de almacenar los datos del jugador como su nombre, la cantidad de filas y columnas con las que está jugando, la cantidad de movimientos realizados, el tiempo, la posición en la que se encuentra la baldosa vacía y, por último, la ruta relativa de la imagen con la que estaba jugando.

7.13. Cronómetro

Utiliza la clase “Timer” de las librerías de “Java Swing”, esta clase se encarga de crear la lógica del indicador del tiempo implementado en la clase “Interfaz”. Se hace uso del “ActionListener”, cuando el usuario presiona la primera baldosa para empezar a jugar, comienza el cronómetro y se actualiza el indicador “etiqueta_tiempo”.

8. Diseño de la interfaz de usuario

La GUI de Java Swing está conformada por dos elementos clave: componentes y contenedores. Los contenedores contienen grupos de componentes y los componentes son los elementos con los que puede interactuar el usuario.

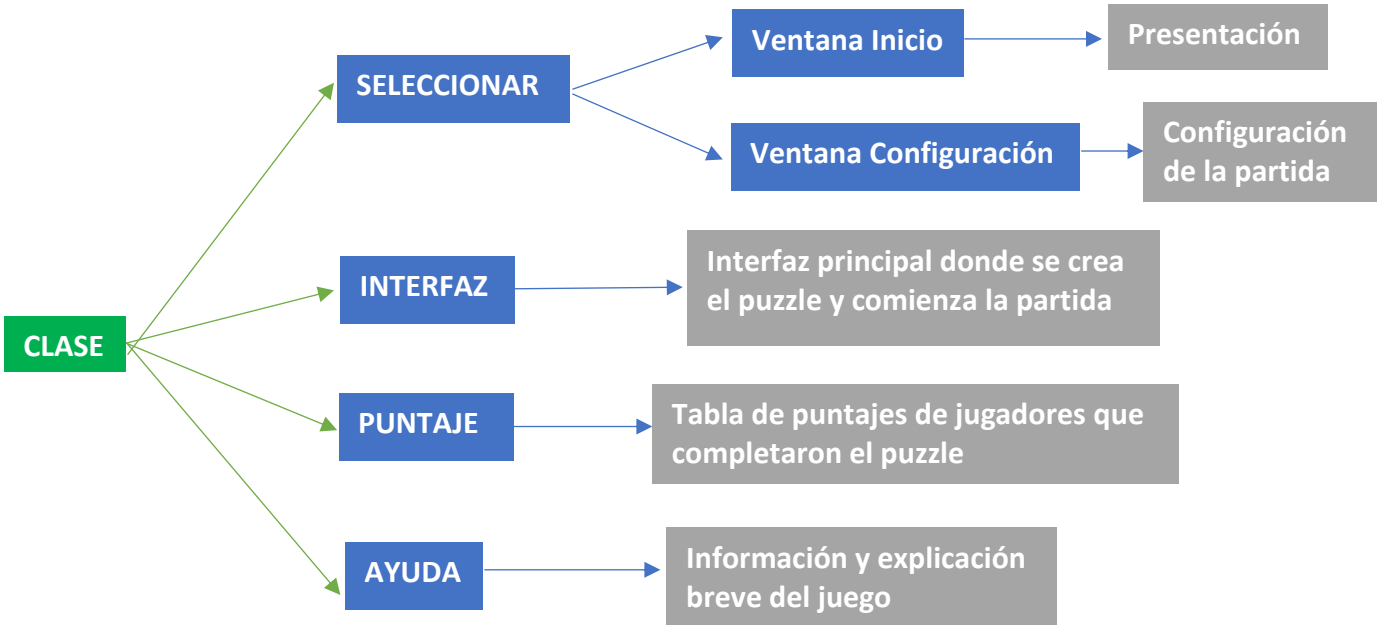
El diseño de la interfaz implementa las funcionalidades de la lógica del juego, entregando al usuario una interfaz gráfica intuitiva al momento de usar la aplicación.

Dentro del desarrollo se utilizaron las librerías de Java Swing en conjunto con el entorno de desarrollo (IDE) Netbeans, que proporcionan herramientas para la creación de interfaces.

Los elementos de la interfaz están incluidos de forma que sea sencilla la interacción con la aplicación y sea visualmente estética, mejorando la experiencia del usuario. Se incluyeron los siguientes elementos: JButton, JTextField, JLabel, JFrame, JPanel y JMenuBar. Estos elementos responden en conjunto a las acciones del usuario a través de la interfaz ActionListener (oyente de acción) y los ActionEvent (eventos de acción).

El diseño de la aplicación está conformado por las ventanas: Inicio, configuración del juego, ventana principal del puzzle, puntajes y ayuda.

Esquema de las interfaces:



9. Justificación del lenguaje

El juego está desarrollado en el lenguaje de programación Java haciendo uso de las librerías "Java Swing", "Java AWT" y "Java IO". Como entorno de desarrollo se utilizó el IDE (Entorno de Desarrollo Integrado) Netbeans, debido a las facilidades que ofrece al momento de crear la interfaz gráfica para el juego e implementar las librerías.

Se eligió para el desarrollo de la aplicación este lenguaje de programación porque permite:

- La creación de la interfaz gráfica (GUI) mediante la implementación de librerías.
- La práctica de la Programación Orientada a Objetos (POO).
- El uso de estructuras de datos como: "ArrayList" y "Pila (FILO)".
- Herramientas para guardar y cargar la partida que proporciona la librería Java IO.

10. Aspectos importantes de la implementación

Dentro del desarrollo del juego Slide Puzzle, se presentan aspectos en la programación que son imprescindibles como:

10.1. La interacción de la lógica del juego con la interfaz gráfica

En un principio la lógica del juego diseñada en código tiene que funcionar en conjunto con la interfaz, que se encarga de mostrar al usuario el comportamiento de la aplicación.

Todas las clases que son parte de la lógica del juego tienen que tener su entorno gráfico integrado de acuerdo a la función que cumplen, para que el usuario de forma intuitiva pueda jugar además de aprovechar complementos del juego como los indicadores de tiempo, pistas y movimientos.

10.2. La creación de las baldosas (JButton) para el puzzle y dividir las imágenes

Las baldosas dentro del juego se generan de acuerdo a la dificultad y la imagen seleccionada por el usuario.

El usuario puede elegir entre una imagen predeterminada que ofrece el juego o una imagen personalizada en los archivos locales del dispositivo. Esta imagen puede dividirse y encajar en las baldosas gracias a los métodos de la clase “SplitImage”.

Se crearon las baldosas del puzzle de la interfaz gráfica de forma manual mediante código para ofrecer al usuario la opción de elegir una dificultad personalizada con el enfoque de ofrecer una solución general del puzzle sin importar la dificultad en la que esté configurado.

10.3. Los movimientos y sus registros

Se diseñó la lógica de los movimientos del puzzle que es algo imprescindible para la funcionalidad del juego. Los movimientos son registrados mediante una pila de datos; estos registros permiten la posibilidad de ordenar el puzzle con una solución general para cualquier dificultad configurada (matriz).

El botón de “Pista” y el indicador de “Pasos restantes” trabajan con la lógica de los registros de movimientos.

10.4. Guardar y cargar partida

Es necesaria la opción de guardar y cargar partida dentro del juego.

El botón de “Guardar Partida” abre una ventana en la que el usuario puede seleccionar la carpeta en la que desea guardar su partida y crear el archivo con un nombre.

Se puede usar el botón de “Cargar Partida” desde la ventana de inicio o al empezar una partida; al cargar partida se abre la ventana del gestor de archivos donde se puede seleccionar la partida previamente guardada.

11. Conclusiones

11.1. Funcionales

El juego en primera instancia cumple con las funcionalidades básicas, cuenta con una interfaz gráfica para el usuario, con opciones para configurar el puzzle, indicadores, la posibilidad de ordenar y desordenar las baldosas de forma automática, y correcto funcionamiento del tablero del puzzle.

Se cumplieron los siguientes requerimientos funcionales:

- Correcto desempeño de las clases y métodos básicos del juego.
- Diseño de la interfaz gráfica.
- El juego encuentra una solución general al momento de resolver cualquier dificultad del tablero definida entre los parámetros.
- Se incluye el botón de “Pista” que resuelve el puzzle por pasos.
- La implementación de indicadores en la partida: Nombre del jugador, tiempo, movimientos realizados, pasos restantes y cuadro del puzzle resuelto.
- Opciones de guardar en la ubicación que el usuario desee y cargar la partida.
- Dificultad personalizada entre los parámetros de 2x2 hasta 10x10 considerando casos de un tablero rectangular como 3x5.
- Múltiples ventanas de partida.
- Tabla de puntuaciones de los jugadores que completaron el puzzle y un buscador.

11.2. De la aplicación

Dentro de la aplicación, el usuario se encuentra con las ventanas de inicio, configuración, y la interfaz de la partida. La interfaz gráfica está implementada de manera que sea sencillo para el usuario interactuar con las opciones e intuitivo al momento de jugar.

La aplicación incluye en la interfaz todas las funcionalidades mencionadas y hace uso de recursos como imágenes, para la personalización del entorno gráfico y la creación del tablero del puzzle.

La aplicación no presenta problemas en tiempo de ejecución, se gestionaron los errores que afectan a la experiencia del usuario mediante pruebas.

11.3. Del trabajo en grupo

Somos 3 integrantes dentro del proyecto de desarrollo del juego; Adan Llanos, Ezequiel Gerstel y Christian Arias. Como grupo consideramos que el trabajo se realizó en equipo, se combinaron los conocimientos de cada uno dando lugar a diferentes funcionalidades en el juego.

Cada uno se enfocaba en un área en específico que faltaba implementar en la aplicación para posteriormente programar en equipo.

Todos los integrantes aportaron sus capacidades, ideas y conocimientos en el área de la programación, y manejo de interfaces. Individualmente cada integrante desarrolló partes clave dentro del proyecto.

Se desarrollaron varias funcionalidades de la aplicación por el trabajo conjunto y en equipo. De la misma forma se gestionaron los errores, en base a pruebas de forma individual para posteriormente dar soluciones en conjunto.

El desarrollo constante y en equipo del proyecto fue el motivo por el cual se concluyó el proyecto de forma satisfactoria con las funcionalidades básicas y adicionales.

12. Bibliografía

Creación del botón para poder cerrar un JTabbedPane
https://icephat43.wordpress.com/2013/10/01/adding-tab-close-buttons-to-a-jtabbedpane-in-java-swing/

Creación de la matriz de botones para el juego
https://www.youtube.com/watch?v=Klc9tLaeunc&t=1064s

Añadir Jpanel a un JTabbedPane
https://www.youtube.com/watch?v=sHxAVTsgNMM&t=473s

Clase que se encarga de recortar las imágenes
https://www.youtube.com/watch?v=pEPKeuLVSio

Como guardar un archivo en los archivos locales del usuario
https://www.youtube.com/watch?v=wMmbCqtZXoE&t=308s

Guardar y cargar un archivo con serialización para el guardado de partidas
https://youtu.be/S8ALWHZWylk

Creación del cronometro
https://www.youtube.com/watch?v=1m0AgWmcawA

Documentación de las librerías de Java Swing (Interfaz)
https://docs.oracle.com/javase/7/docs/api/javax/swing/package-summary.html

Documentación de las librerías de Java IO (Guardar y Cargar)
https://docs.oracle.com/javase/7/docs/api/java/io/package-summary.html

Implementación de ventanas emergentes JOptionPane
https://serprogramador.es/programando-mensajes-de-dialogo-en-java-parte-1/
http://www.edu4java.com/es/java/joptionpane-showmessagedialog-showinputdialog.html

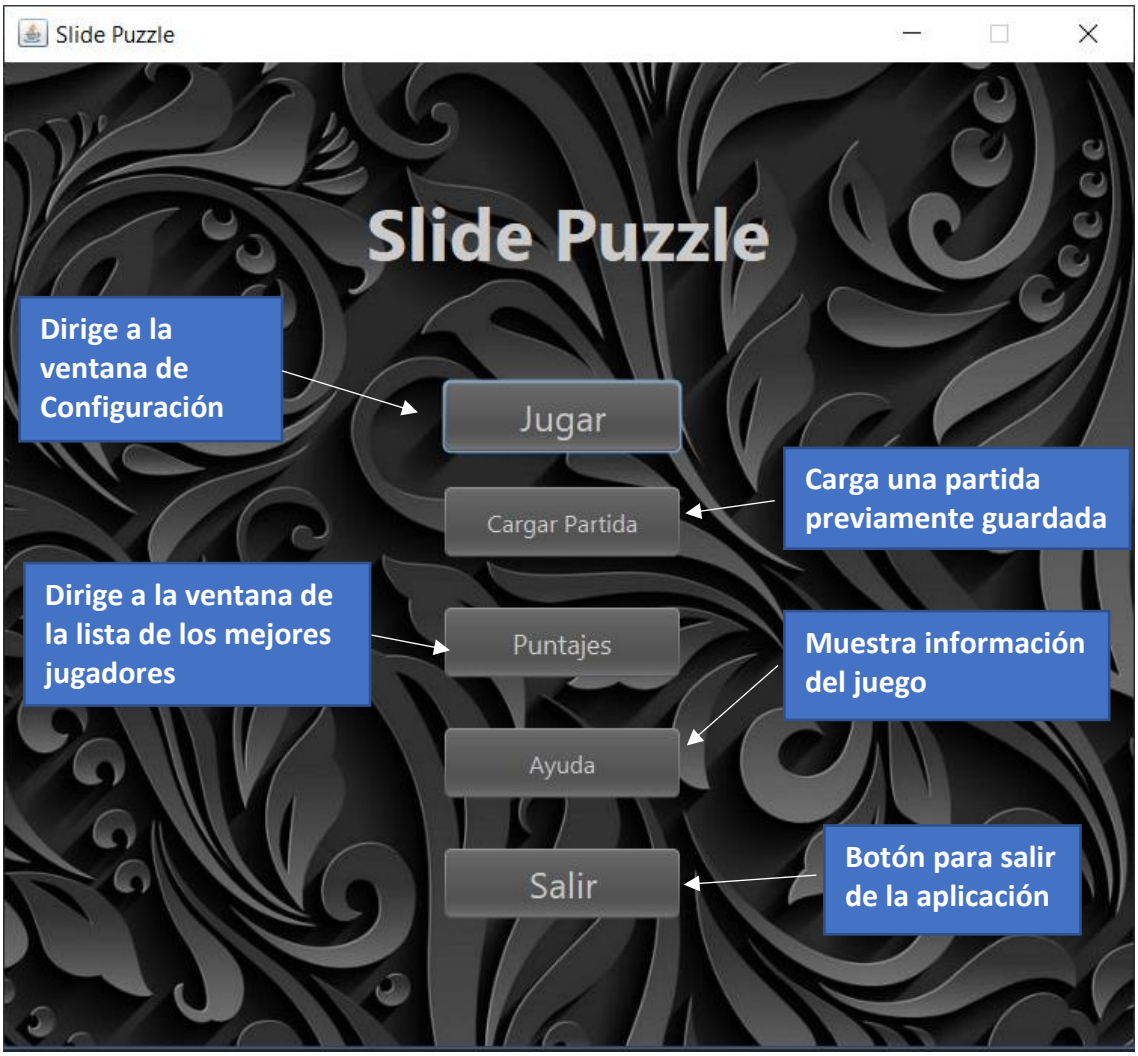
Diseño de la interfaz
https://www.youtube.com/watch?v=XvaaZTVbjps&list=PLeZ7bixkO9gFLHqgxF7VwHGGQWmD04072&index=2

Uso del método “random()” de la clase Math para desordenar el puzzle
https://www.educative.io/edpresso/how-to-use-the-mathrandom-method-in-java

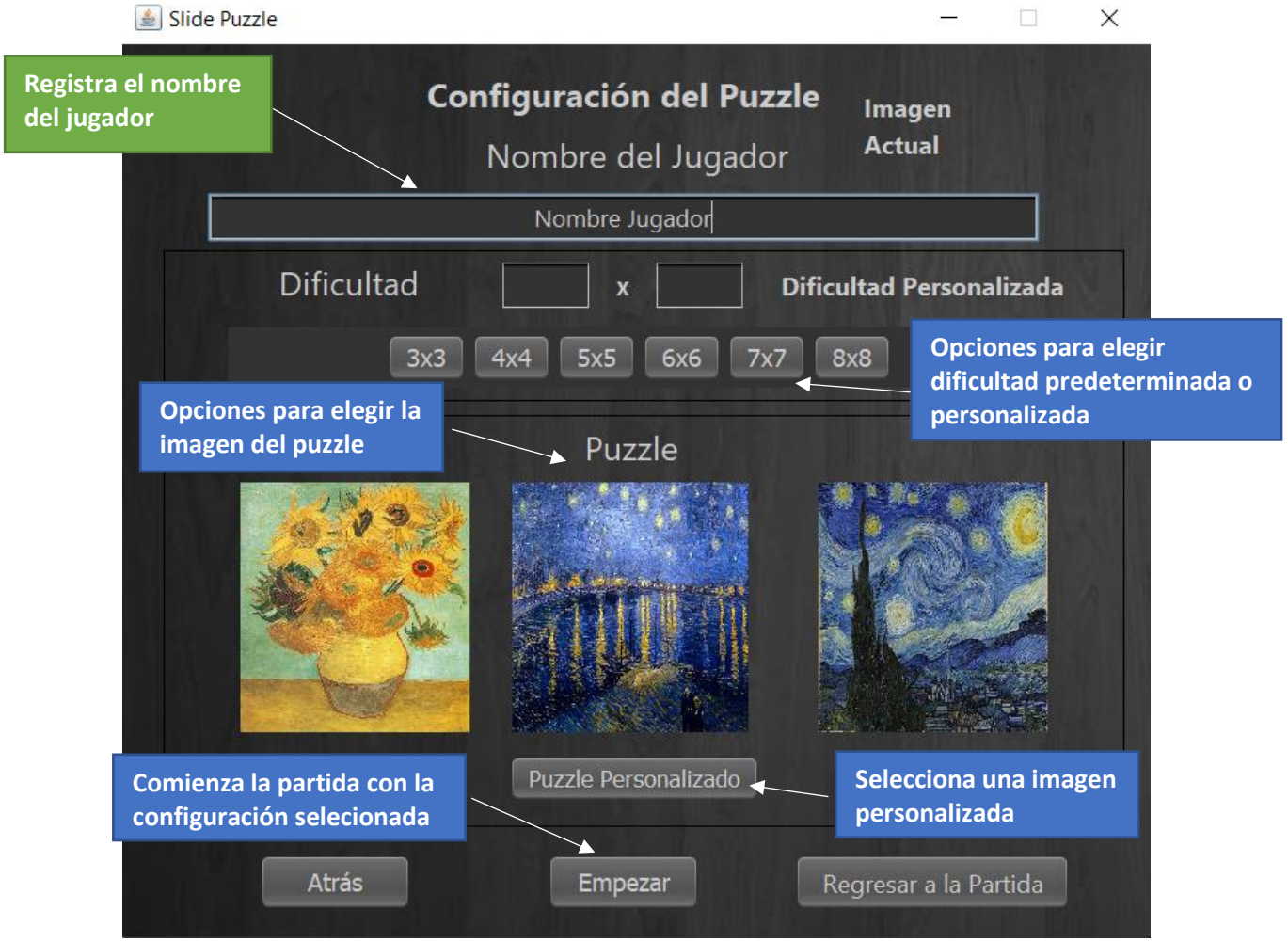
13. Anexos

A. Manual de Usuario e Interfaz Gráfica

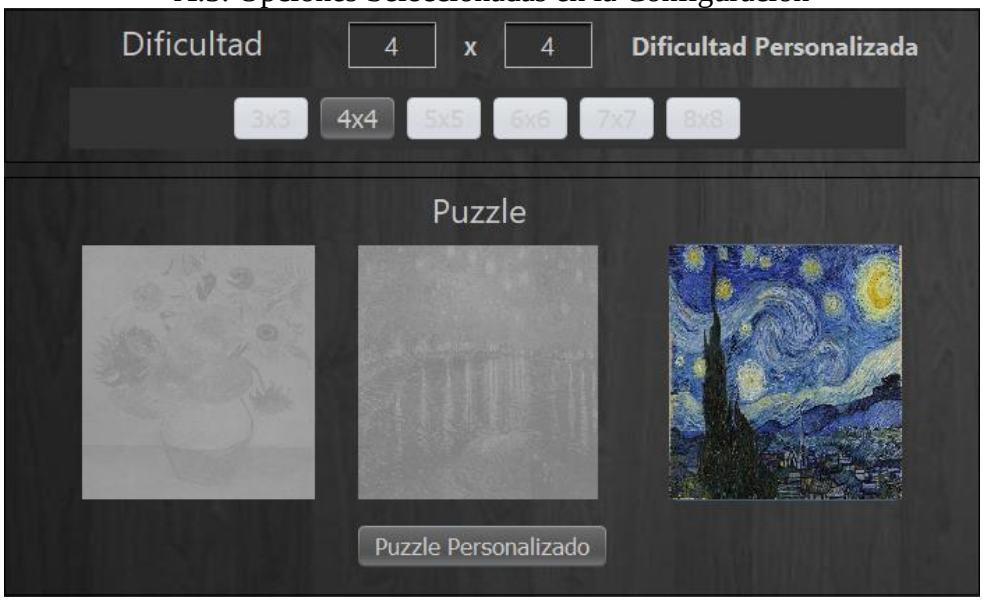
A.1. Ventana de Inicio (Presentación)



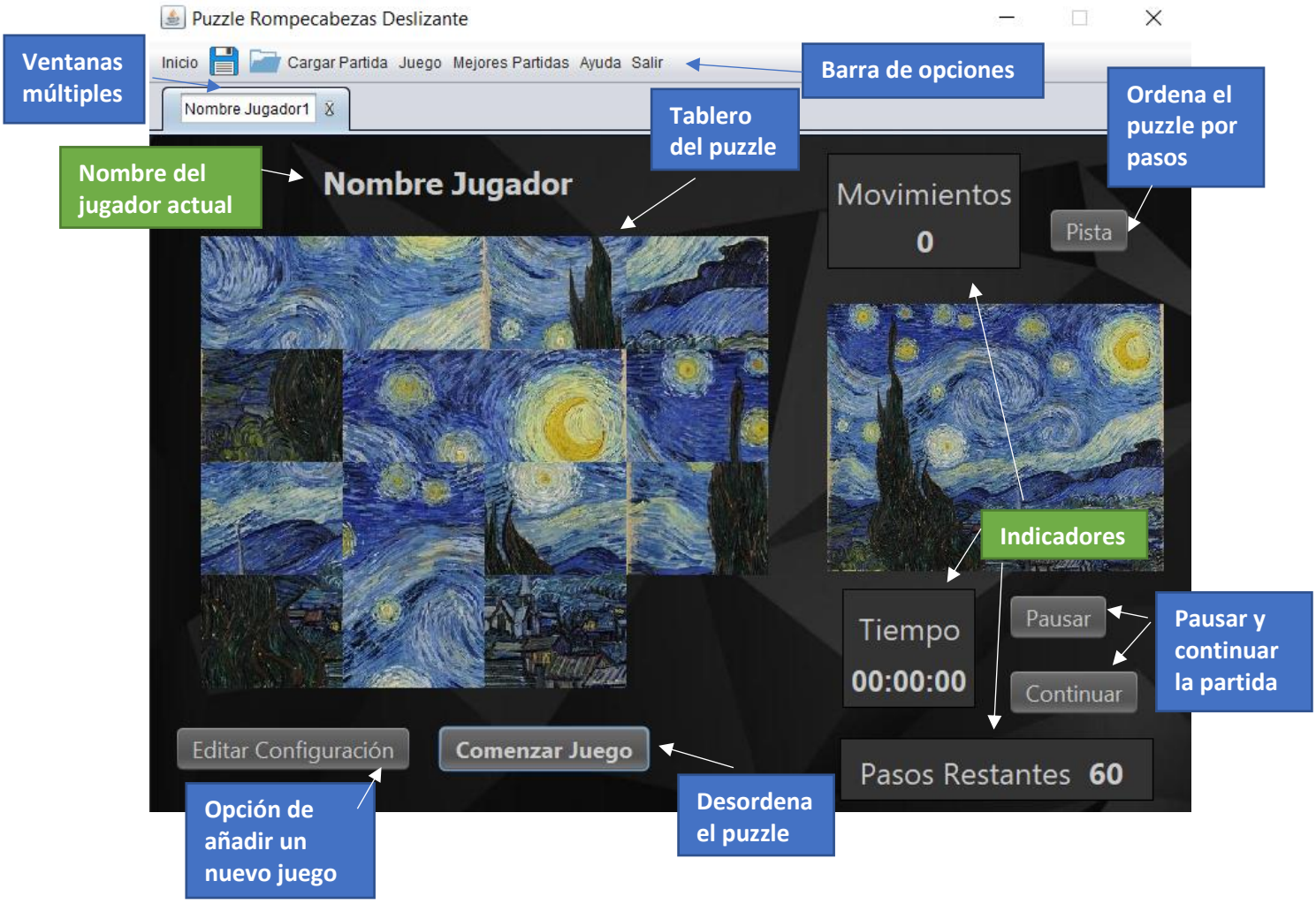
A.2. Ventana de Configuración de la Partida



A.3. Opciones Seleccionadas en la Configuración

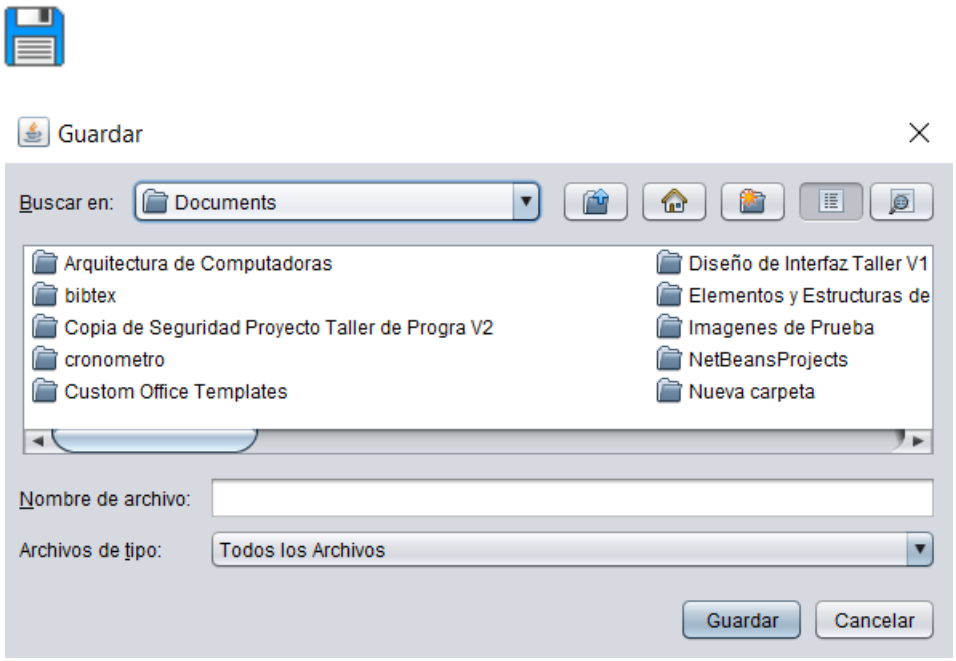


A.4. Ventana de la Interfaz

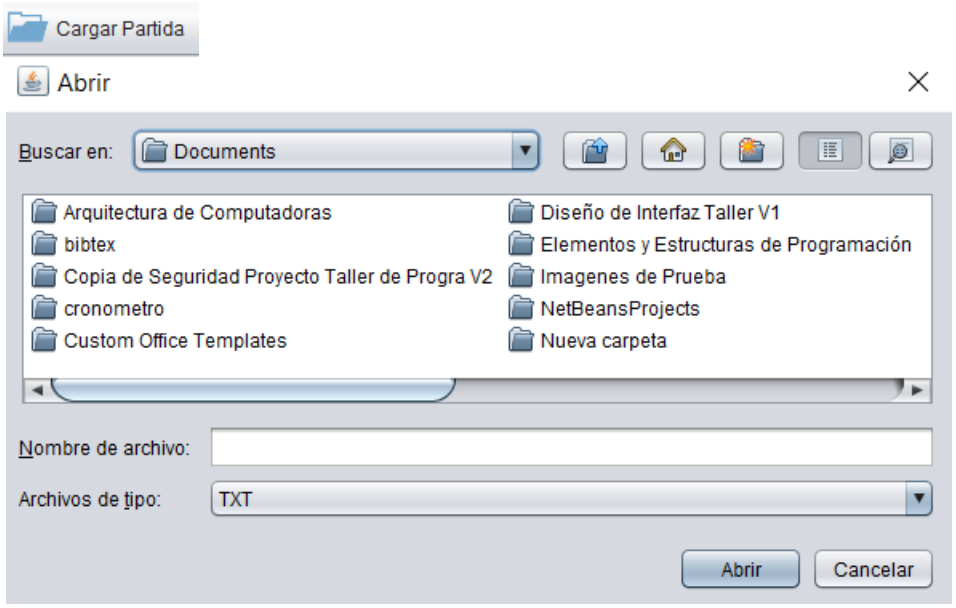


A.5. Barra de Opciones del Juego

A.6. Ventana Guardar Partida



A.7. Ventana Cargar Partida



A.8. Tabla de Puntajes Registrados (Ranking)

A.9. Ventana de Ayuda (Información)