# **El lazarillo de tormes**

**Juego: “Memorion”.**

# --- Desarrolladores ---

# Guillermo Manso García

# Jesús Roncero García

# Índice

1 – Análisis de requisitos

2 – Definición de clases

3 – Funcionalidades extra

4 – Errores y solución de los mismos

5 – Diagrama

## 1 - Análisis de requisitos

|  |  |  |
| --- | --- | --- |
| **Requisito** | **Obligatorio** | **Realizado y/o observaciones** |
| Una interfaz gráfica funcional y multiplataforma. | **SI** | **Sí** |
| Una interfaz en lengua inglesa | **SI** | **Sí** |
| Modo de juego definido anteriormente. | **SI** | **Sí** |
| La temática del juego será sobre el “Lazarillo de Tormes” Se deberá generar un fichero .jar que se entregará junto con el proyecto | **SI** | **Sí** |
| Inserción e implementación de un Menú. | **SI** | **Sí** |
| Necesidad de que una partida acabe cuando se complete el panel | **SI** | **Sí** |
| Un contador de tiempo en segundos. | **SI** | **Sí** |
| Un contador de tiempo especificando minutos y segundos. | **NO** | **Sí** |
| Un contador de jugadas. | **SI** | **Sí** |
| En caso de que la segunda carta levantada no sea pareja de la primera, se deberá dejar un tiempo visible las cartas destapadas, y posteriormente de forma automática, se volverán a ocultar. | **SI** | **Sí** |
| Un listado con las puntuaciones de la gente. | **SI** | **Sí** |
| El listado de las puntuaciones se ordena indicando en primer lugar los jugadores que lo han resuelto en menor tiempo, y en caso de empate en el menor número de jugadas. | **NO** | **Sí** |
| Se deberá mostrar un cuadro de Dialogo modal en el que aparezca información acerca del autor de la aplicación. | **SI** | **Sí,**  **en la opcion del menu de salir, esto ocurre siempre.** |
| En el cuadro de diálogo anterior se puede mostrar la foto del jugador la cual se pedirá en el mismo sitio a la vez que se introduce el nombre. | **NO** | **Sí,**  **la información del jugador se solicita antes de cada partid, junto otras característica.** |
| Deberán aplicarse los aceleradores de teclado que se muestran en la demo, teniendo su funcionalidad implementada (excepto el de guardar y cargar partida) | **SI** | **Sí** |
| Se podrá guarda el estado de una partida, salir del programa y volver a recuperar la partida grabada. Si se desea se puede implementar la opción de grabar varias partidas y cargar la que decida el usuario. | **NO** | **Sí,**  **solo se permite guardar una partida, se puede cambiar, (cambiando una variable) que cuando carge la partida, esta partida que cargue, se elimine.** |
| Se deberá poder elegir 3 niveles distintos para jugar. | **SI** | **Sí** |
| Se podrá introducir un huevo de pascua que haga alguna función diferente. | **NO** | **Sí,**  **un juego de picar en globos** |
| Se deberá introducir una pantalla de carga inicial, en lo que carga el juego | **NO** | **Sí,**  **existen 3, uno que solo se muestra en la 1º ejecución del programa, otro que se muestra cada vez que cambia de vista y otro que se ejecuta cuando se sale del programa** |
| El usuario podrá configurar con qué imágenes desea jugar. | **NO** | **Sí,**  **puede seleccionar entre 3 y/o elegir una de su equipo** |
| La aplicación deberá matar todos los hilos y procesos al cerrarse. | **SI** | **Sí** |
| La aplicación podrá tener una opción de resetear los datos, de tal forma que elimine las estadísticas de las partidas ganadas y todos los archivos que se hayan creado. | **NO** | **Sí,**  **en el menú.** |
| Para realizar los guardados de datos y las cargas se utilizará la técnica que presenta Java de serialización. | **NO** | **Sí** |
| Los movimientos de las cartas deberán realizar una breve transición ( no tiene que ser la misma que la del video) | **SI** | **Sí** |
| Incluir un icono personalizado para la aplicación, que se vea en los Sistemas operativos Windows, Linux y MacOS. | **NO** | **Sí** |
| Si el usuario desea cerrar la aplicación en mitad de una partida, se deberá mostrar un mensaje al usuario preguntándole si está seguro de lo que quiere salir. | **NO** | **Sí** |
| Se podrán manejar los movimientos de las jugadas con las teclas del teclado. | **NO** | **Sí,**  **pulsando antes en un botón o cambiando de pantalla y volviendo a elegir la del juego** |
| Se deberá añadir al proyecto una documentación asociada con los puntos que se indica en este documento. | **SI** | **Sí** |
| Se podrán implementar mejoras a la aplicación siempre que no contradigan las funcionalidades expuestas. | **NO** | **Sí** |

## 2 – Definición de las clases

Hemos dividido el proyecto en distintos paquetes orientados a la agrupación de los distintos objetivos de trabajo, en un paquete hemos agrupado todas las vistas, en otro todos sus controladores, y en otro paquete distinto hemos alojado la lógica, la vista principal, el main, y demás funcionalidades mínimamente independientes

### Paquete de Vistas

VistaSplash - es la clase encargada de mostrar el splash, la pantalla de carga que se muestra entre los cambios de vista y al arrancar el programa, se le pueden asignar distintos tiempos de carga a cada pantalla dependiendo de la funcionalidad que le queramos dar, para el inicio por ejemplo, está establecido en 10 segundos

Vprincipal – Es la vista que contiene el menú de nueva partida, estadisticas, el acceso a la información de los desarrolladores y cargar la partida anterior, además del easter egg en un botón oculto en la parte central de la pantalla

Vmenu – Es la vista encargada de añadir el menu superior al programa

Vingreso – Es la vista previa al juego, en ella se asigna un nombre, si no se modifica queda predeterminado “Nickname”, se elige un avatar, puede modificarse y elegir uno personalizado clicando en el el campo de texto que pone “Avatar”

Se elige un tema, el tema va a asignar las cartas elegidas para jugar

Se elige una dificultad, asigna el numero de cartas con el que se va a jugar

Vlista – Es la vista que contiene las estadisticas, contiene un listado con todos los datos además de una barra con scroll para cuando los datos acumulados superan el alto de la pantalla

Vjuego – en esta vista estan contenidas las cartas, además de un contador que va sumando segundos y tres botones, uno para parar la partida, otro para guardar el estado de la misma, y otro que está deshabilitado hasta que se acaba la partida, que nos lleva a la vista de las estadísticas

VdialogoMod – contiene los datos de los desarrolladores, así como un boton hacia la vista principal y un logo de trinitarias que si se pulsa lleva a la página del colegio

Carta – es la clase encargada de generar las cartas en la vista juego

### Paquete Controladores

ContrDialogoMod – contiene toda la lógica derivada de la vista DialogoMod

ContrIngreso – contiene toda la lógica derivada de la vista VIngreso

ContrJuego - contiene toda la lógica derivada de la vista VJuego

ContrLista - contiene toda la lógica derivada de la vista VLista

ContrMenu - contiene toda la lógica derivada de la vista VMenu

ControladorPascua - contiene toda la lógica que maneja el huevo de pascua

ControladorPrincipal - contiene toda la lógica que controla la vista principal

### Paquete LazarilloTormes

Historial - aloja el array que contiene todas las estadísticas

HuevoPascua - se encarga de generar el huevo de pascua con los globos y los “mensajes de error”, al clicar en 10 globos, se cierra

Lógica - contiene toda la lógica del programa y se relaciona con los controladores

LogicaPascua - contiene la lógica del huevo de pascua

PartidaGuardada - es la clase encargada de guardar en los ficheros la partida

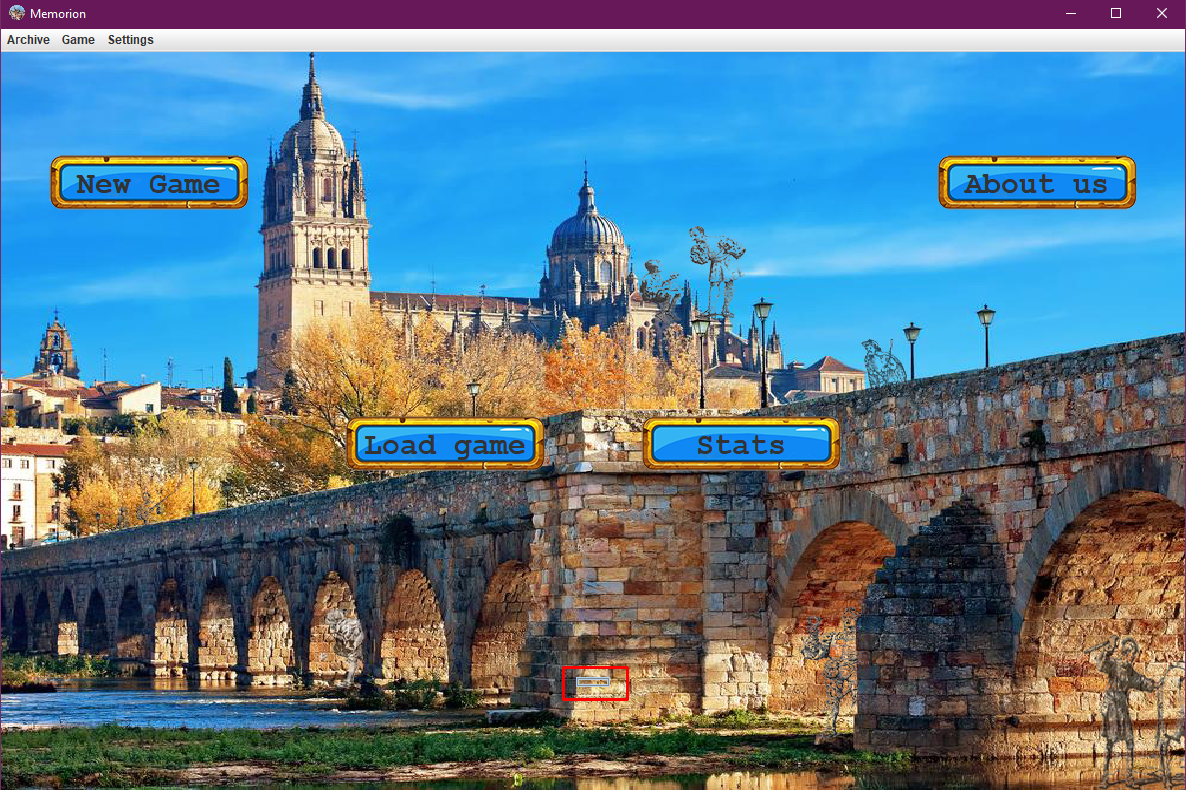
Sonido - se encarga de todos los sonidos del juego

TrabajoLazarilloMain - es el main, crea la vista

Vista - es la clase que se encarga de cambiar entre todas las vistas

## 3 - Funcionalidades extra:

* Se implementó sonido durante la ejecución de la aplicación, teniendo 3 sonidos por cada acción y eligiendo uno aleatorio cada vez que se ejecute una acción.
* Opción en el el menu, en “settings” para poder apagar el sonido, durante la ejecución del programa.
* Se implementó una vista para visualizar el contenido del fichero de estadísticas, no teniendo un máximo extenso pero no infinito, gracias a la aplicación de un ScrollPane.
* Se implementó en el menú funciones extra como:
  + Nueva partida rápida, los valores se ajustan por defecto.
  + Nueva partida personalizada, los valores se ajustan según el usuario
* Se han implementado múltiples animaciones en múltiples fondos de pantalla.
* Huevos de pascua:
  + Minijuego, haciendo click en uno de los ladrillos del puente de la vista principal, se genera un juego que trata de hacer click en 10 globos, al hacer esto, este se cierra automáticamente.



* + Al acabar la partida muy rápido, en la pulsación entre las dos últimas cartas, todas las cartas se comprimen en la vista de juego y queda un efecto muy curioso y divertido
  + Se implementó la opción de que el usuario pueda escoger su propio avatar en la vista de ingreso de datos. Haciendo click en la palabra “Avatar”.

## 4 - Problemas surgidos durante la creación

Durante la creación de la aplicación del surgieron varios problemas:

* **Trabajar desde el mismo fichero,** necesitábamos trabajar sobre el mismo fichero, en netbeans y que este se fuese actualizando con un control de versiones. La solución fue usar la herramienta “GitHub” con el cual hemos realizado este proyecto. se debe de tener en cuenta al utilizar esta herramienta, que si se modifica el mismo archivo (clase) él las mismas líneas a la vez este dará un error cuando la 2º persona intente actualizar correctamente la versión.
* **Finalizar la vista splash**, este problema viene causado por compaginar el cierre de un panel con otro, ya que la la vista splash contiene Timers y si realizamos sleep o wait, se dormían también los timers. La solución fue realizar otro método para y añadir la vista, que ejecutase cuando el splash terminase
* **Ajustar los elementos al tamaño de la pantalla en el splash**, el problema era causado ya que el panel del splash era nulo y ya que el logo de “Trinitarias” rotaba, sobreescribiendo el paint(), los campos eran ajustados por coordenadas. La solución fueron múltiples reglas de 3, para que se ajustara a la pantalla , tanto el logo como los componentes.
* **Posición de los elementos al tamaño de la pantalla,** en todas las vistas, excepto en la vista Splash, tienen un layOut, la solución para que los componentes se ajustarán a la pantalla y en una posición (relativa) fue usar el “gridbaglayout”
* **En la vista juego se podían ver más de dos cartas al mismo tiempo,** durante la ejecución del juego si se realiza muy rápido se llegaban a ver más de dos cartas al mismo tiempo, esto era causado por el tiempo de la animación, la solución fue en el TimeTask que se encargaba de dar la vuelta a las cartas cuando no eran la misma asignarle un tiempo para que se volviese a ejecutar(tiempo que tardaban en darse la vuelta las cartas y que una vez realizada la 2 ejecución del TimeTask este se parase.
* **En las estadísticas al incluir una imagen podía dar varios errores,** al permitir que el propio usuario pudiese elegir un una imagen de su equipo, y se pasaba el historial de estadísticas , esta ruta en el 2 equipo no existiría y daria nulo, produciendo errores, la solución fue crear una condición que en el caso de que fuese nulo o que fuese una ruta absoluta y el archivo no existiera, asignase uno por defecto.
* **El usuario tiene la opción de acceder a documentos sin que estos existan,** esto da error puesto que son nulos , la solución fue comprobar que existía, si no existía mostrar un mensaje y no cargarlo, en el caso de cargar la partida, en el caso de las estadísticas, mostrará un panel vacío
* **Guardar y cargar partida, serialización,** para serializar la partida, todos los elementos han de ser serializables, cuando cargamos la partida guardada, algunos de los elementos no se cargaban correctamente. La solución: tener una clase serializable que guardase los datos, en lógica, y que cagarse esta clase al juego.
* **Inserción de imágenes, sonidos, y archivos en el archivo.jar,** el problema era causado ya que en el archivo.jar las rutas no son las mismas que ejecutandolo por netbeans, en el netbeans las rutas parten desde /src/. La solución fue a la inserción de las rutas partiesen de “this.getClass().getResouce(RUTA)” en el caso de las imágenes y sonidos.
* **Modificación y creación de ficheros desde el archivo.jar,** el problema viene causado ya modificar, crear y eliminar ficheros dentro del mismo archivo durante la ejecución da errores en la ejecución, la solución fue crear estos archivos que se necesitasen en la ruta donde se localiza el archivo.jar para esti de uso “new File(fichero).getAbsolutePath();”

## 5 - Diagrama de clases