Manual Técnico Arkanoid

REALIZADO POR:
Laura Ivonne Shahidinejad Martínez
#00365919
Rodrigo José Flores Chévez
#00053714
Oscar Alexander Cornejo Valsse
#00223019

CONTENIDO

Manual técnico Arkanoid ---------- 1

Aspectos generales ---------- 3

Objetivo del documento ---------- 3

Descripción general ---------- 3

Software utilizado ---------- 3

Modelos utilizados ---------- 4

UML diagrama de clases ---------- 4

Diagrama relacional normalizado de base de datos utilizada ---------- 5

Conceptos técnicos ---------- 6

Implementación de interfaz gráfica ---------- 6

Manejo de clases en modelo ----------6

Plataforma base ---------- 6

Nomenclaturas ---------- 7

Abreviaturas ---------- 7

Eventos y Excepciones ----------8

Eventos ---------- 8

Excepciones ---------- 8

ASPECTOS GENERALES

Objetivo del documento

El objetivo de este documento es dar a conocer la forma en que se realizó este proyecto, el manejo de todo lo aprendido durante el ciclo en la materia y como fue implementado explicando el diseño de este.

Descripción general

Para la creación del software de este proyecto se utilizó el MVC, lo que se conoce como Modelo - Vista - Controlador. El programa esta basado en el juego de los 80s llamado “Arkanoid”, basado en Breakout de Atari de 1976, el cual consiste en la destrucción de bloques por medio de una barra y una bola.

Software utilizado

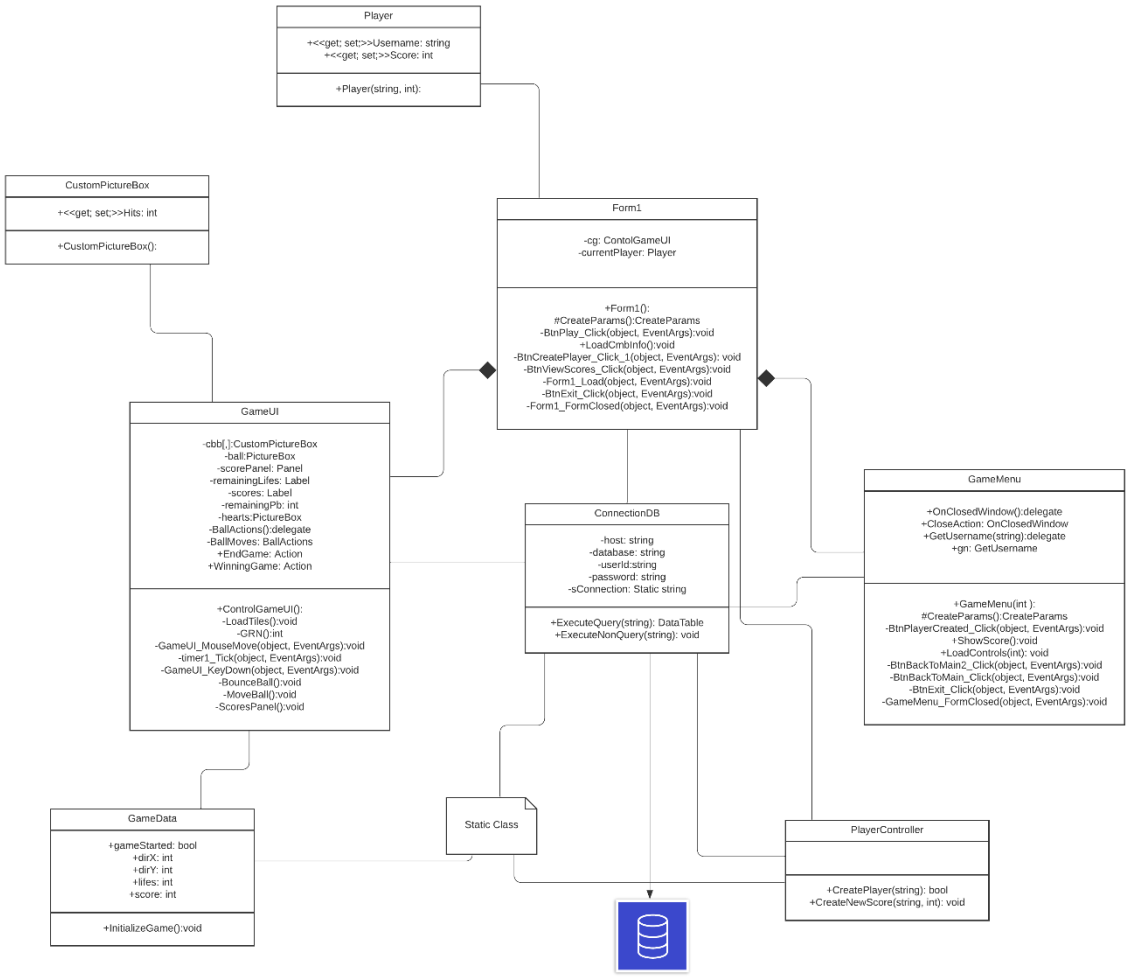
Para la creación de este programa se utilizó Jetbrains Rider 2019.3.4, en conjunto con PostgreSQL para la creación de la base de datos. Complementos adicionales para Rider utilizados ha sido Npgsql para la conexión, LucidChart y draw.io para los diagramas.

MODELOS UTILIZADOS

UML Diagrama de clases

El diseño del código está basado en el diagrama de clases siguiente:

Se deja el enlace de la imagen en Google Drive para una vista más detallada: https://drive.google.com/file/d/1WN_owHaeTGyDIvW-O1H82s_EjnGNx4AZ/view



Explicando el diagrama de clases, se puede elegir a un usuario, el cuál ejecutara el juego y se puede cambiar al comienzo de cada partida.

ConnectionBD es la clase enlace con la base de datos creada de manera local para poder realizar consultas y almacenar datos, en específico, datos de usuarios y puntajes.

El Form1 o formulario principal se inicia con un menú, en el cual se puede seleccionar al jugador, escoger la opción para jugar (la cual carga un UserControl en el cual se

encuentra el juego), agregar a un nuevo jugador, ver el top 10 de los puntajes obtenidos hasta el momento y una opción para salir del juego.

El PlayerController se encarga de agregar nuevos usuarios y de ingresar nuevos puntajes por medio de consultas de la clase ConnectionBD.

La clase Player se creó para almacenar datos específicos de cada jugador y se utiliza en el Form1.

GameData posee variables que se utilizan en el desarrollo del juego, tales como las vidas, el puntaje, si el juego ha comenzado o no, etc.

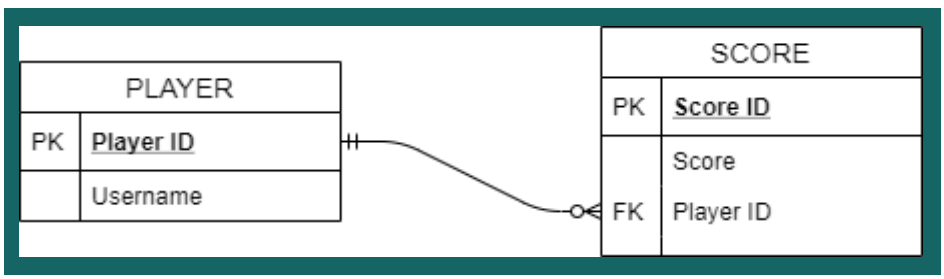
La clase CustomPictureBox posibilita el uso de los golpes, lo cual se utiliza en el UserControl en donde se genera el juego.

El GameMenu permite visualizar una ventana dependiendo del botón que se seleccione en el Form1, en una se puede agregar a un usuario y en la otra ver el top 10 de los puntajes realizados por los jugadores.

Presionar el botón “Jugar” de Form1 dirige al jugador hacia el UserControl de ControlGameUI, el cual carga todos los datos gráficos para poder jugar. Al iniciar el juego se poseen 3 vidas, se encuentran bloques con distintos límites de golpes, se muestra el puntaje en tiempo real y las vidas que el jugador posee en el momento, al finalizar, se almacena el puntaje en la base de datos solamente si el jugador gana la partida.

Diagrama relacional normalizado de base de datos utilizada

En este diagrama se visualiza la manera en que se decidió estructurar la base de datos utilizada en el programa, la cual cuenta con dos tablas, una para usuarios y otra para los puntajes:



CONCEPTOS TÉCNICOS

Impletación de interfaz gráfica

La interfaz gráfica de este programa está compuesta por 2 ventanas o formularios, la principal siendo Form1 y contando con un **diseño de mesa** (tableLayoutPanel en inglés), con botones y una **caja combo** (Combo Box en inglés) para seleccionar al jugador. Dos de esos botones se encargan de cambiar el formulario a otro formulario secundario llamado GameMenu, en el cual se puede agregar a un usuario o mostrar puntajes. Otro botón de Form1 se encarga de cargar un **control de usuario** (User Control en inglés) en el cual se desarrolla el juego como tal.

Form1.cs ----------(Form principal)
GameMenu.cs ---------(Form secundario)
ControlGameUI.cs ---(User Control)

Manejo de clases en modelo

Para el manejo de este programa se cuenta con las siguientes clases:

ConnectionBD.cs
GameGata.cs
CustomPictureBox.cs
Player.cs
PlayerController.cs

Plataforma base

Sistema Operativo	**Multiplataforma**
Tecnologías	**JetBrains Rider 2019.3.4**
Lenguaje	**C#**
Gestor de BD	**PostgreSQL**

NOMENCLATURAS

Abreviaturas

Para los elementos del entorno gráfico del proyecto, se utilizaron las siguientes abreviaturas:

Button	btn
PictureBox	pb
TextBox	txt
TableLayoutPanel	tlo
ComboBox	Cmb

EVENTOS Y EXCEPCIONES

Eventos

El evento implementado para los formularios GameMenu y Form1 fue:

- PlayerController.cs
  Registra datos de un jugador incluyendo el puntaje.

Excepciones

Se utilizaron cinco excepciones en el desarrollo del juego, las cuales son auto explicativas, su nombre indica su función. Las excepciones son la siguientes:

EmptyUsernameException.cs

ExceededMaxCharactersException.cs

NoRemainingLifesException.cs

OutOfBoundsException.cs

WrongKeyPressedException.cs