STICK CLASH

PROYECTO
DESARROLLO DE APLICACIONES MULTIPLATAFORMA

Realizado por Alejandro Romero Contreras

Curso 2022/23

Sumario

Estudio preliminar

Forma	“Stick Clash” en un juego de modalidad versus. Dos jugadores luchan en una arena uno contra otro con el objetivo de golpearlo para derrotarlo y así alzarse con la victoria.
Mecánica	- Cuando tu o tu rival es golpeado por una bala o cae al vacío pierde, y tu ganas la ronda. - En cada ronda solo tienes una vida. - Al ganar una ronda ganas 1 stick, al acabar se guardan tus sticks ganados. - Las partidas serás de 1v1 en modo local. - El jugador 1 jugará con el teclado, y el invitado con gamepad.
Contexto	En un mundo futuro donde todo es aburrido se ha inventado una forma de divertirse donde dos jugadores luchan a muerte de forma amistosa. Claramente no mueres y puedes jugar cuantas veces quieras en atractivos escenarios impulsados por la competitividad.
Arte	El videojuego ofrece una perspectiva 2D, con un arte pixelart, y diseños sencillos. Los colores serán simples, sobre todo para que los personajes sean diferenciables del fondo. Para acabar los escenarios tendrán temáticas por ejemplo una obra o un campo.
Target	El juego será para todos los públicos, aunque se espera una moda estadística en torno a jugadores de entre 14 – 22 años. El juego saldrá para Windows, Linux y MacOs.

Descripción, justificación del proyecto y su alcance.

El proyecto presentado podría definirse como una aplicación de escritorio destinada al ocio y entretenimiento, también llamada videojuego. Dispondrá de una serie de mecánicas que permitirá al usuario interactuar con los elementos en pantalla para llegar a conseguir unos objetivos definidos y a su vez de divertirse mientras “juega”. Más adelantes se explicarán esas mecánicas y objetivos.

A nadie le gusta usar aplicaciones aburridas por lo que la finalidad del juego es hacer que los usuarios que lo jueguen pasen un rato entretenido con sus amigos, ya que es un juego para 2 jugadores.

El juego está destinado a cualquier usuario, de cualquier edad o género, aunque en base a las estadísticas de edad sobre los usuarios que consumen videojuegos, se espera que la edad de jugadores oscile entre 14 y 30 años.

Para diferenciarnos ofreceremos el juego de forma gratuita, y se pretende ofrecer una buena experiencia de juego, habrá diferentes personajes dentro del juego que fomentaran la rejugabilidad.

Stack tecnológico

Para el proyecto se utilizarán diversas tecnologías para desarrollarlo.
El motor gráfico Unity junto con el IDE Visual Studio.
El lenguaje de programación usado será C#.
Base de datos mySQL.
Contenedor en Docker para hostear al server mySQL.
Photoshop para la edición de Assets de imagen.

Objetivos

El objetivo del proyecto es crear un videojuego con la finalidad de llegar al máximo de usuarios posibles, entretener a los mismos, que sea accesible a cualquier usuario con interfaces simples, y por carácter personal la entrada al mundo del desarrollo de videojuegos.
Otro objetivo de este videojuego es de ser un puente de acceso para personas que nunca han probado algún juego o simplemente no conocen este mundo, a que lo prueben y la descubran.

Valoración de alternativas

Duck Game

Controlas a patos que luchas con diversas armas en un escenario

Nidhogg

Juego de duelos 1v1 donde debes vencer a tu oponente para avanzar hacia a la derecha de la pantalla y llegar hasta el final

Jump ‘n Bumb

Juego en el que manejas a conejos con el objetivo de saltarles encima y derrotarlos. Este juego fue creado para MSDOS.

Requisitos del sistema

Requisitos funcionales:

Sistema de login por usuario y contraseña, sin confirmar el acceso no es posible acceder.
Sistema de registro para nuevos usuarios, el usuario no podrá repetirse.
En el menú la opción crear partida alojará al jugador 1 y se mantendrá a la espera de que aparezca un jugador 2.
En el menú la opción buscar partida alojará al jugador 2 y buscará una partida creada.

El personaje se moverá por teclado con las teclas de dirección y disparará con la tecla “Z”, no serán modificables.

Requisitos no funcionales:

La base de datos debe estar siempre operativa, ya que si no el juego no es accesible.
Base de datos en mySQL alojado de forma gratuita en https://www.freemysqlhosting.net/
Ordenador con SO Windows 7 o superior, procesador Dual Core y tarjeta gráfica.

Requisitos de interfaz:

Interfaz con colores básicos basado en paletas de colores para desarrollo de interfaces.

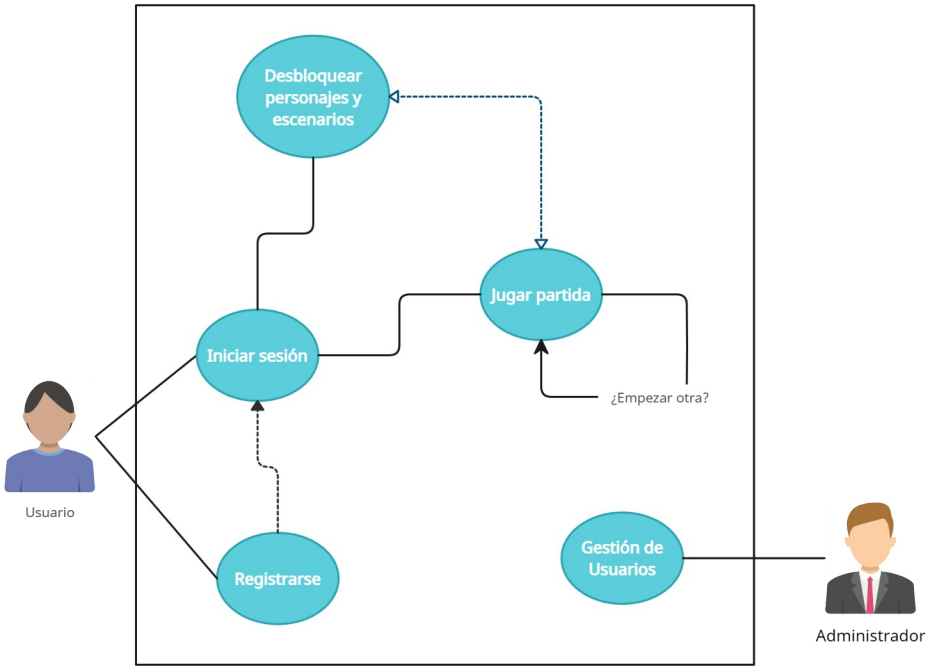
Deben diferenciarse los botones de los menús con facilidad y que sean los menos posibles.
Compatible con pantallas en resolución 16:9.
Personajes jugables con colores y contornos, junto con los fondos usados en los escenarios, que hagan que se distingan correctamente.

Mejoras contempladas

- Añadir el 4 personaje jugable.
- El personaje de invitado podrá seleccionarse.
- Modo online.
- Añadir modo aventura.
- Rediseño de los personajes.
- Poder salir en medio de una partida.
- Añadir efectos visuales y de sonido.
- Habrá una barra para controlar el volumen de la música y efectos de forma general.

Casos de uso

Diagrama de casos de uso



Descripción de casos de uso

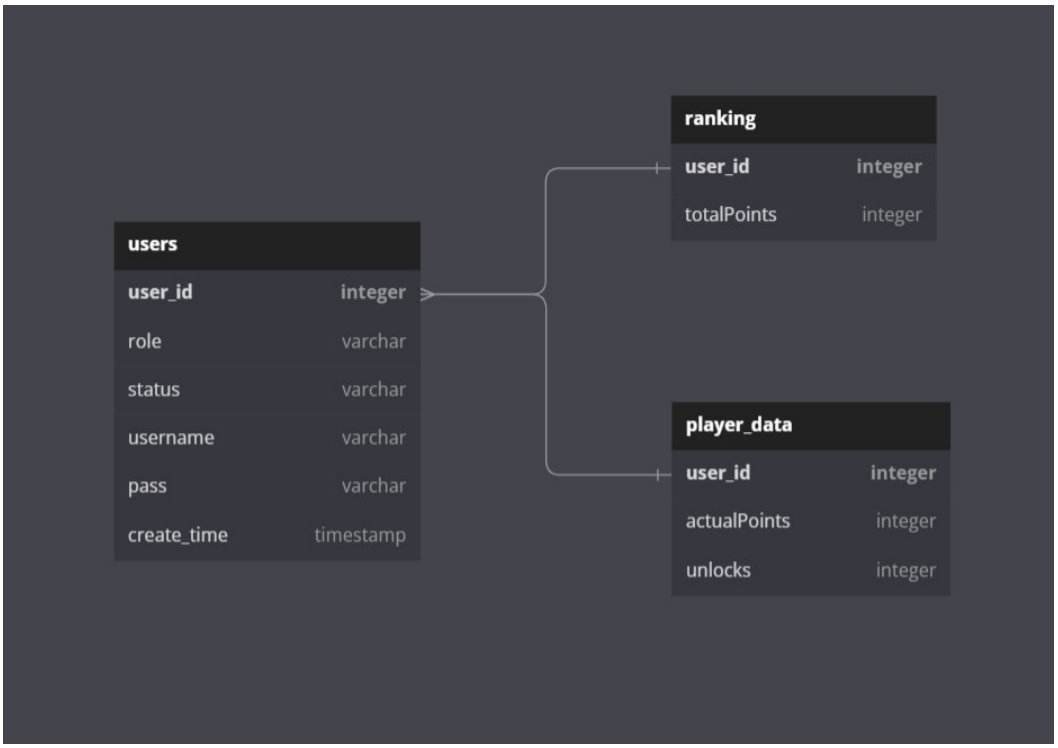
CU - 01		
Versión	V0.2	
Dependencias	Sistema de login/registro funcional	
Precondición	Un nuevo usuario sin registrar	
Descripción	Intentar loguearse y registrarse tras error de usuario no encontrado	
Secuencia Normal	**Paso**	**Acción**
	1	Se rellena los campos y se presiona sobre iniciar sesión.
	2	El sistema revisa en la base de datos que el usuario no existe.
	3	El sistema muestra mensaje de error de que el usuario no existe.
	4	El usuario presiona sobre “Nuevo?”.
	5	Rellena los campos de registro usuario, contraseña y repite contraseña, posteriormente presiona “Registrarse”.
	6	El sistema comprueba que el usuario no existe, posteriormente lanza la inserción del nuevo usuario a la base de datos.
	7	El sistema da acceso al menú principal con el nuevo usuario ya iniciada la sesión.
Postcondición	El usuario tiene acceso completo al juego.	
Excepciones	1.1	El usuario no rellena algún campo
	1.2	El sistema advierte de que no puede haber ningún campo vacío.
	5.1	El usuario no rellena algún campo
	5.2	El sistema advierte de que no puede haber ningún campo vacío.
	6.1	El usuario ya existe y el sistema lo indica como mensaje de error.

CU - 02

Versión	V1	
Dependencias	Selección de personaje y carga de este en el escenario de juego.	
Precondición	Iniciada sesión	
Descripción	Seleccionar un personaje y empezar a jugar.	
Secuencia Normal	**Paso**	**Acción**
	1	Se carga el menú de jugador.
	2	Se hace click en “Jugar”.
	3	El sistema carga los menús de selección de personaje
	4	El usuario presiona sobre uno de los 3 personajes disponibles.
	5	El sistema guarda el usuario seleccionado en cada momento.
	6	El usuario presiona “Empezar”
	7	El sistema carga el escenario de juego.
	8	El sistema rescata el usuario guardado y elige aleatoriamente el personaje invitado.
	9	El sistema instancia ambos personajes y la ronda de juego.
Postcondición	El jugador puede mover los personajes y jugar.	
Excepciones	1.1	El usuario presiona cerrar sesión y vuelve a la pantalla de inicio de sesión.
	4.1	El usuario vuelve atrás al menú principal de jugador.
	6.1	El usuario vuelve atrás al menú principal de jugador, se guarda el último personaje seleccionado.

Modelo y diseño de la base de datos

Modelo de la base de datos, sujeto a futuros cambios en base a ampliaciones:



El script para la creación de la base de datos se encuentra en el archivo *“stick_clash_bd.sql”*

URI de conexión:
mysql://root:examplepassword@home.netindio.synology.me:3306/stick_clash

Script de la base de datos

```
-- phpMyAdmin SQL Dump
-- version 5.2.1
-- https://www.phpmyadmin.net/
--
-- Servidor: 127.0.0.1
-- Tiempo de generaciÃ³n: 11-06-2023 a las 18:02:51
-- VersiÃ³n del servidor: 10.4.28-MariaDB
-- VersiÃ³n de PHP: 8.2.4

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Base de datos: `stick_clash`
--
CREATE DATABASE IF NOT EXISTS `stick_clash` DEFAULT CHARACTER SET utf8mb4
COLLATE utf8mb4_general_ci;
USE `stick_clash`;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `player_data`
--

DROP TABLE IF EXISTS `player_data`;
CREATE TABLE `player_data` (
  `user_id` int(3) NOT NULL DEFAULT 0,
  `actualPoints` int(3) NOT NULL DEFAULT 0,
  `unlocks` int(2) NOT NULL DEFAULT 0 COMMENT 'cada digito se representa
individualmente (decenas los encenarios y las unidades los personajes).'
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Volcado de datos para la tabla `player_data`
--

INSERT INTO `player_data` VALUES(2, 7, 0);
INSERT INTO `player_data` VALUES(3, 14, 0);
INSERT INTO `player_data` VALUES(4, 13, 0);

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `ranking`
--
```

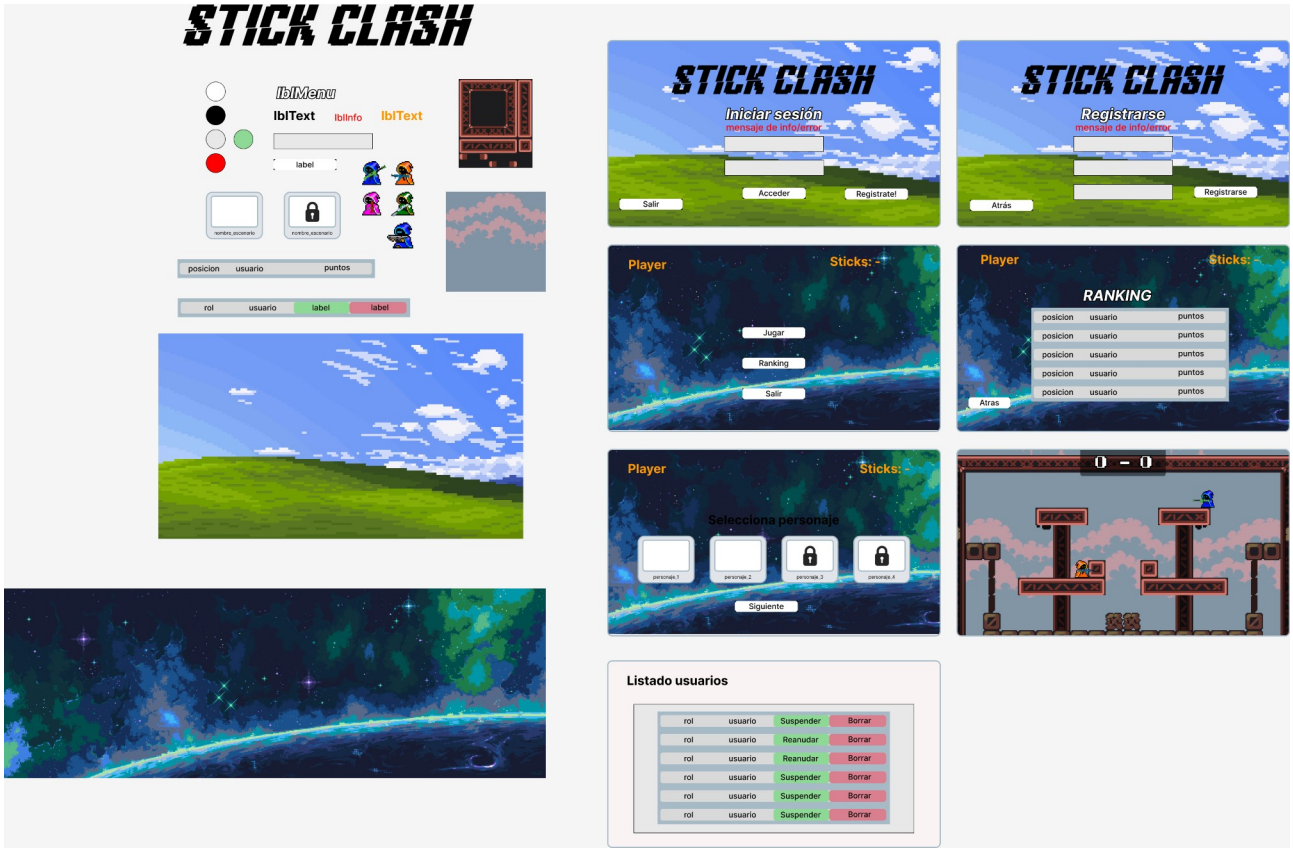
```
DROP TABLE IF EXISTS `ranking`;
CREATE TABLE `ranking` (
  `user_id` int(3) NOT NULL,
  `totalPoints` int(3) NOT NULL DEFAULT 0
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Volcado de datos para la tabla `ranking`
--

INSERT INTO `ranking` VALUES(2, 7);
INSERT INTO `ranking` VALUES(3, 25);
INSERT INTO `ranking` VALUES(4, 13);

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `users`
--

DROP TABLE IF EXISTS `users`;
CREATE TABLE `users` (
  `user_id` int(3) NOT NULL,
  `role` varchar(10) NOT NULL DEFAULT 'player',
  `status` varchar(9) CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL
DEFAULT 'activated',
  `username` varchar(15) NOT NULL,
  `password` varchar(20) NOT NULL,
  `create_time` timestamp NOT NULL DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Volcado de datos para la tabla `users`
--

INSERT INTO `users` VALUES(1, 'admin', 'activated', 'admin', 'admin123',
'2023-06-07');
INSERT INTO `users` VALUES(2, 'player', 'activated', 'player1', '1234',
'2023-06-07');
INSERT INTO `users` VALUES(3, 'player', 'activated', 'player2', '1234',
'2023-06-07');
INSERT INTO `users` VALUES(4, 'player', 'activated', 'alex', '1234',
'2023-06-11');

--
-- Ã□ndices para tablas volcadas
--

--
-- Indices de la tabla `player_data`
--
ALTER TABLE `player_data`
  ADD PRIMARY KEY (`user_id`);

--
-- Indices de la tabla `ranking`
--
ALTER TABLE `ranking`
```

```
  ADD PRIMARY KEY (`user_id`);

--
-- Indices de la tabla `users`
--
ALTER TABLE `users`
  ADD PRIMARY KEY (`user_id`);

--
-- AUTO_INCREMENT de las tablas volcadas
--

--
-- AUTO_INCREMENT de la tabla `users`
--
ALTER TABLE `users`
  MODIFY `user_id` int(3) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=5;

--
-- Restricciones para tablas volcadas
--

--
-- Filtros para la tabla `player_data`
--
ALTER TABLE `player_data`
  ADD CONSTRAINT `player_data_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES
`users` (`user_id`) ON DELETE CASCADE ON UPDATE CASCADE;

--
-- Filtros para la tabla `ranking`
--
ALTER TABLE `ranking`
  ADD CONSTRAINT `ranking_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES
`users` (`user_id`) ON DELETE CASCADE ON UPDATE CASCADE;
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

Diseño

Libro de estilos y prototipo de la interfaz

Tanto el libro de estilos y el prototipo de interfaz pueden verse en: https://www.figma.com/file/lZIN24QMybpejbmNzt5oyF/Stick-Clash?type=design&node-id=0%3A1&t=ja4VRYqFGUpEQkcB-1



Manual de Usuario

Menús de acceso

Nada más iniciar el juego nos encontraremos con la pantalla de título, tras hacer click en cualquier lugar nos llevará a la pantalla de acceso de usuarios.

En esta pantalla podremos acceder con nuestro usuario y contraseña, se mostrará un mensaje rojo de advertencia si no se encuentra el usuario, por ejemplo.
Si el acceso es correcto el juego cambiará al menú de jugador, con la sesión iniciada.
Cabe destacar que si el usuario introducido tiene el rol *admin*, el juego se redirigirá a la pantalla de gestión de usuarios, y no al menú de juego.

Tenemos un botón *"Nuevo?"* para poder crear un usuario y contraseña si no disponemos de uno ya.

Tenemos por último botón que finaliza la aplicación.

Si en el anterior menú hemos pulsado sobre *"Nuevo?"* nos aparecerá la siguiente pantalla. En ella podremos registrar un nuevo usuario, introduciendo un nombre de usuario, contraseña y confirmación de contraseña.
Si el usuario ya existe o las contraseñas no coinciden, se indicará en un mensaje de error.

Si rellenamos los campos correctamente y pulsamos sobre *"Registrarse",* el juego cambiará al menú de jugador con el nuevo usuario.

Podemos pulsar en *"Atrás"* para ir a la pantalla de acceso de usuarios.

Menú administrador

En esta pantalla disponemos de una tabla dinámica donde se muestran todos los usuarios con rol *player.*
Muestra el número identificador de usuario, el username, el estado y un apartado de accesiones sobre ese usuario en concreto.

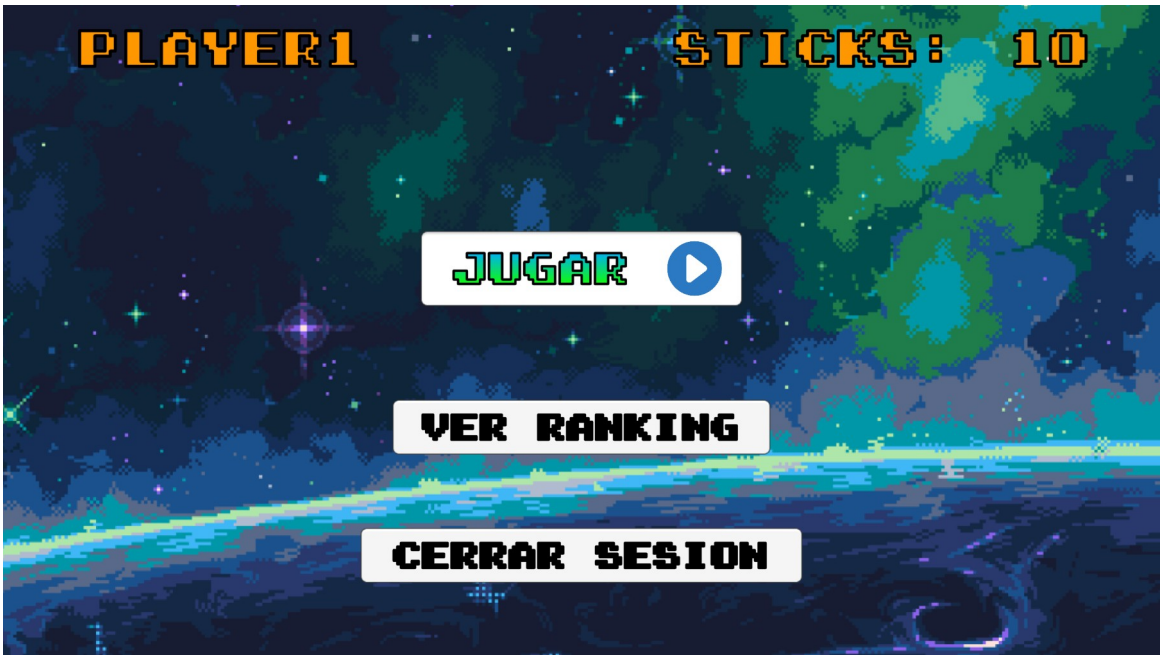
El estado *activated* indica que el *player* puede acceder a su sesión del juego.
En cambio si está en estado *suspended,* el jugador NO podrá acceder.

El botón "*Cambiar estado",* cambia entre estos dos estados.

Con el botón *"Eliminar"* podremos eliminar permanentemente un usuario, para evitar borrados accidentales se debe confirmar la acción.
"SI" → Elimina el usuario
"NO" → Cancela la acción (volvería a aparecer el botón *"Eliminar"*);

Menú de jugador

Al acceder nos encontramos varias opciones, se puede observar que arriba carga tu nombre de usuario y puntos actuales.



Al pulsar sobre ver ranking podemos ver todos los usuario ordenados por sus puntos totales.

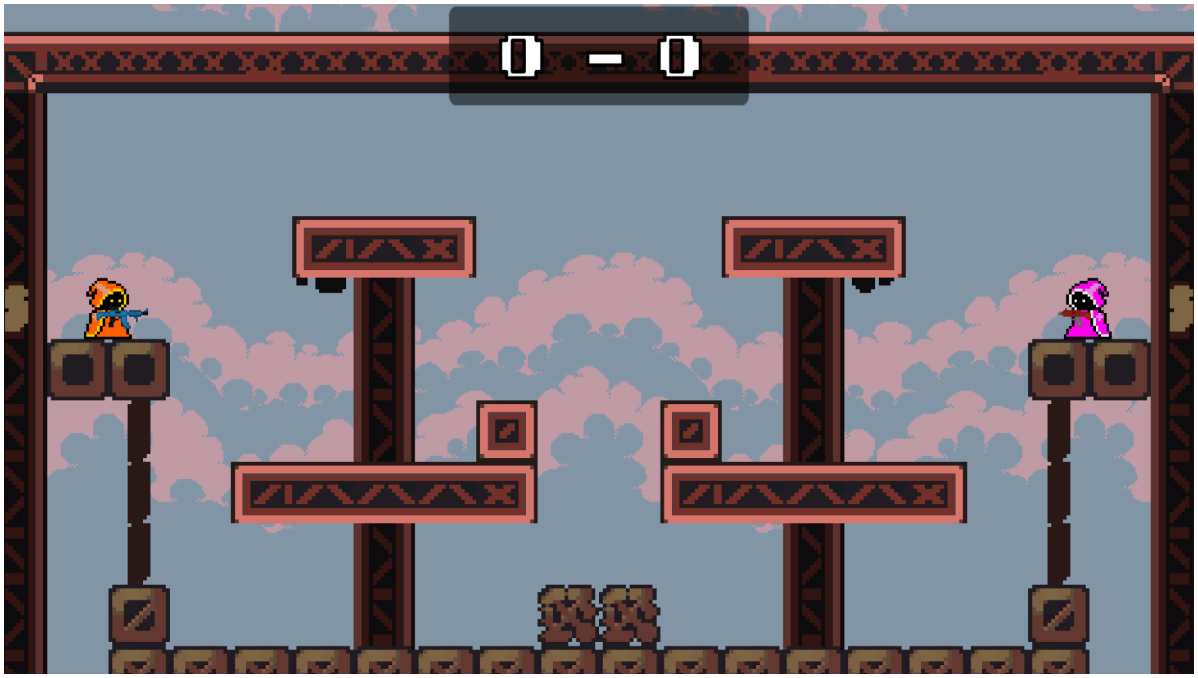
Al presionar en jugar aparece el menú de selección de personaje.
Tras seleccionar uno de los personajes (el 4 aún no está implementado) aparece el botón siguiente para empezar y empezará el juego.

El jugador invitado tendrá un personaje aleatorio.

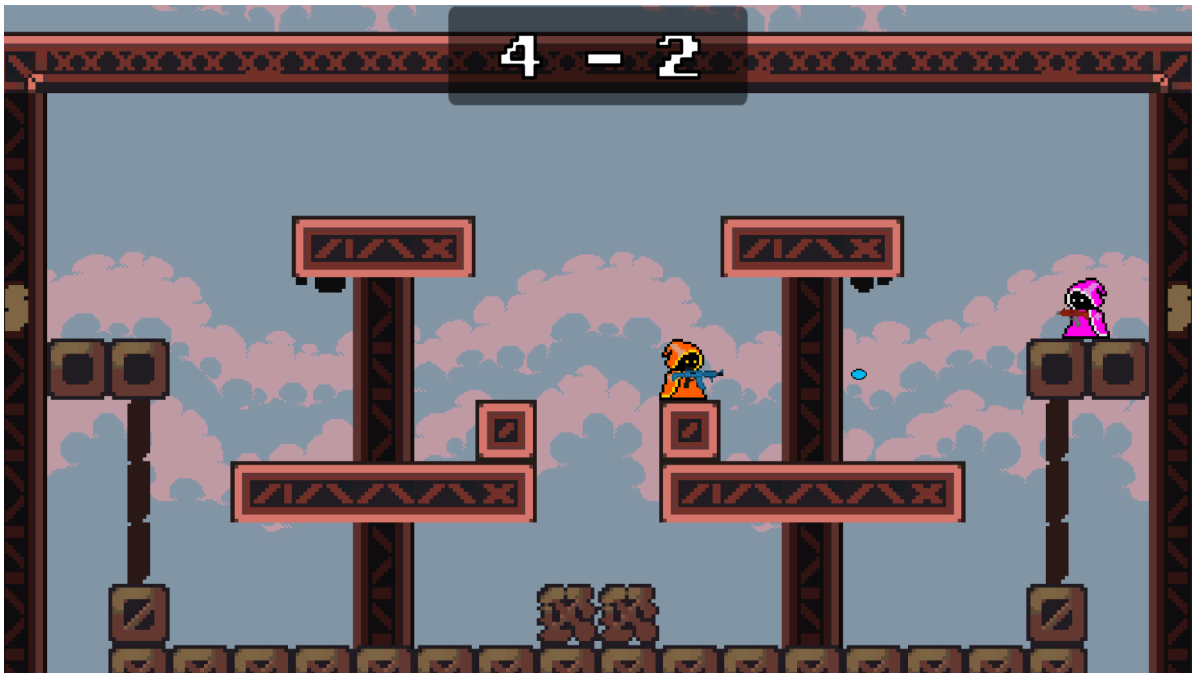
Pantalla de Juego

Al comenzar tendremos el siguiente escenario.

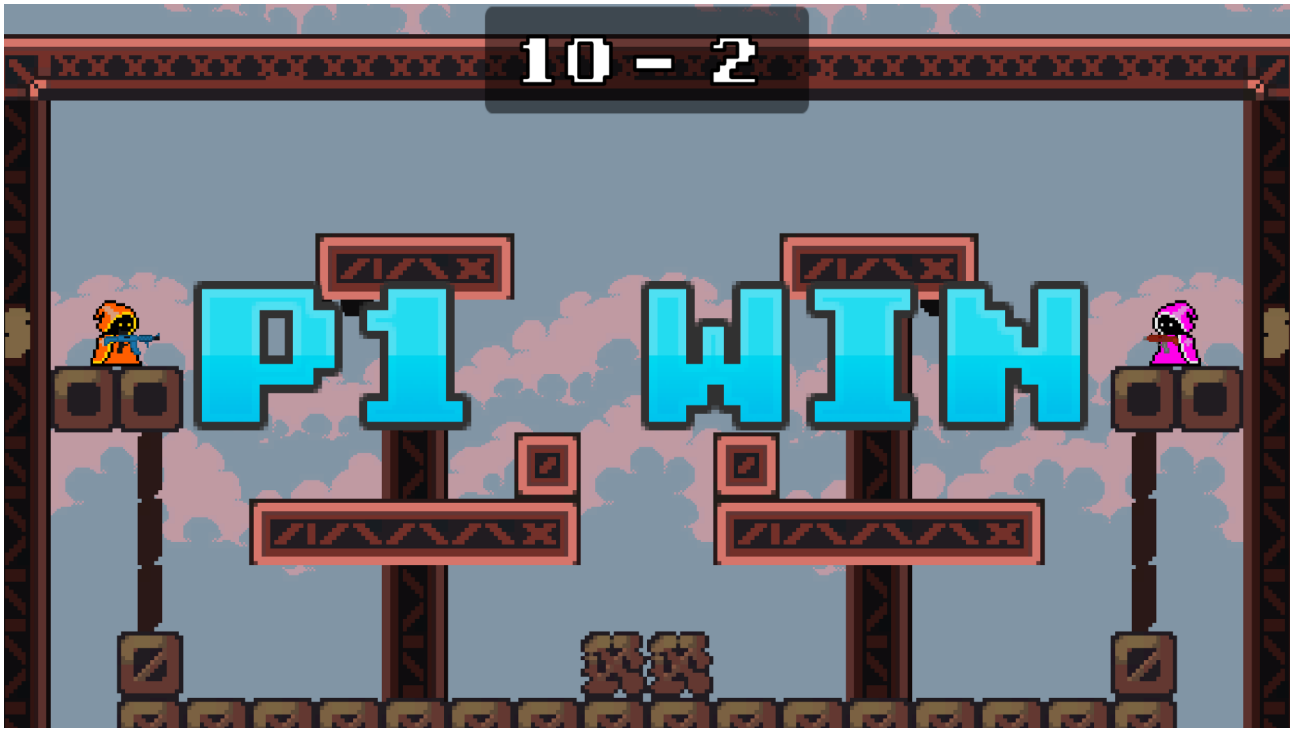
El jugador 1 (izquierda) se controla con las flechas de dirección, dispara con “x” y salta con “z”.
El jugador 2 (derecha), el cual tiene un personaje aleatorio de entre los disponibles, se controla con un mando o gamepad.



Cuando un personaje cae al vacío o es golpeado por una bala el otro jugador suma un punto, y se reinicia la ronda.



Cuando un jugador llega a 10 puntos gana la partida (aparece un mensaje indicando el ganador en pantalla).



Tras completar la partida, el sistema carga el menú de jugador donde puedes ver tus puntos acumulados, ranking o empezar otra partida con otro personaje.

Errores / problemas encontrados

MySqlconnector

A la hora de crear la conexión con la base de datos mysql es normal que el IDE no tenga las dependencias (referencias en c#). La cuestión fue añadirlas a este para que compilara.
Cuando añadía las referencias a VS parece ser que lo detectaba bien y compilaba en VS pero no es Unity. Cuando cerraba VS y volvía abrirlo desaparecía. Lo instalé a través del explorador del propio VS, modificando la configuración de la herramienta de VS llamada NuGet por si esta el que lo borraba, y configurando la referencia con la ruta del dll en el archivo que recoge estas mismas, pero nada, seguía borrándose.

La solución que encontré es que el proyecto, incluidas las referencias, las gestiona el proyecto en Unity. Simplemente hay que añadir en Assets el dll y se solventaría. Al hacerlo el proyecto compila automáticamente y desaparecen los errores.

Nota: Se ha usado la versión del conector 6.10.9, ya que la última (8.) no era compatible.

Colisiones del personaje no esperadas

Por problemas de unity las colisiones básicas del mapa producen puntos en el que el personaje se queda pillado. Para ello había que recurrir a otro tipo de colisiones de tilemap y generar una malla de colisiones.

Sistema de rondas

He encontrado bastante complejidad el sistema de rondas, es necesario comprobar muchas cosas para que funcione todo bien tales como la muerte del jugador, colisiones y muertes, cuando instanciar o destruir los personajes, reiniciar la ronda y algunas cosas más.

A raíz de esto he modularizado más el código y es más fácil acceder a entender el código.