**Daniel Espinosa Garcia**

**Proyecto RA1 RA2 - Primera Fase con los CE**

**RA1**

## c) Se han identificado las tecnologías de desarrollo de aplicaciones para dispositivos móviles.

Como podemos observar a lo largo del tiempo han existido múltiples sistemas operativos para los cuales se desarrollaron diferentes tecnologías para desarrollar aplicaciones.

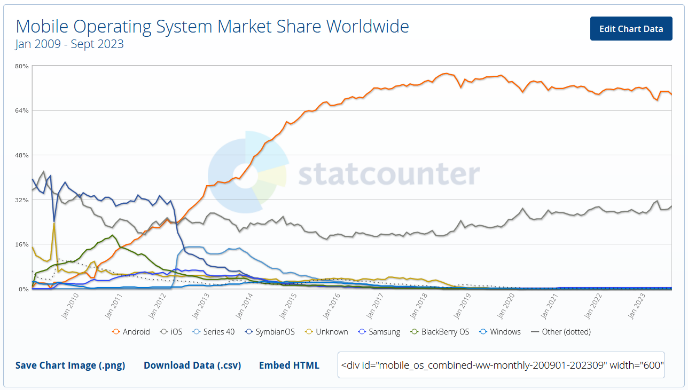


Ilustración 1 https://gs.statcounter.com/os-market-share/mobile/worldwide

En la actualidad el mercado está dominado por los sistemas operativos de Android y iOS.

Estos dos sistemas operativos (SO) utilizan tecnologías distintas para la programación de aplicaciones en el lenguaje nativo de cada SO



**Swift**: Para desarrollo en iOS, se utiliza este lenguaje de programación ofrecido por Apple para el desarrollo de aplicaciones que funcionen en sus plataformas: iOS, macOS, tvOS, watchOS y iPadOS.

Dibujo en blanco y negro

Descripción generada automáticamente con confianza media**Kotlin:** Para desarrollo en Android, este lenguaje permite crear apps nativas de alta calidad y gran rendimiento en todas las plataformas soportadas por Android: smartphones, televisores, relojes, etc.

La principal desventaja de estos dos sistemas es que el desarrollo de aplicaciones en los lenguajes nativos no permite portar las aplicaciones entre sistemas operativos por lo cual se requiere duplicar el trabajo.

Para evitar esta duplicidad de trabajo se desarrollaron otras tecnologías para poder portar a ambos SO

**Ionic:** En primer lugar, tenemos ionic, el framework probablemente más popular que nos permite desarrollar aplicaciones multiplataforma. Nos ayuda a construir apps con HTML, CSS3 y Javascript empleando todo el potencial de las tecnologías de desarrollo web.

**Flutter:** Seguidamente, encontramos Flutter, una herramienta desarrollada por Google para el desarrollo de aplicaciones multiplataforma. Está basada en Dart y aunque se presentó en 2015, no se lanzó hasta 2018. Una gran ventaja frente a otras soluciones es que la aplicación desarrollada puede funcionar hasta en 4 plataformas: iOS, Android, Web y Desktop.

**React Native:** Es un framework creado por Facebook en 2015 para el desarrollo de apps multiplataforma.

Al igual que Ionic, también utiliza Javascript, pero lo hace bajo el pretexto de orquestar controles de interfaz de usuario (UI) específicos de la plataforma.

## d) Se han instalado, configurado y utilizado entornos de trabajo para el desarrollo de aplicaciones para dispositivos móviles.

Para desarrollar aplicaciones móviles instalamos Unity y Android Studio ambos IDE permiten la creación de apps.

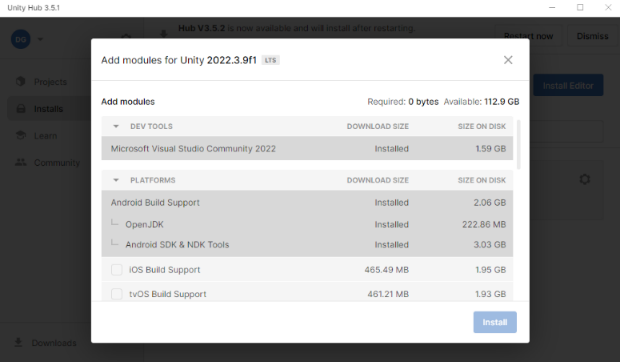


Ilustración 2 configuración de Unity Hub para permitir la exportación a SO Android

Algunos de los IDEs más populares son:

**Android Studio**: es una popular herramienta de programación recomendada por Google. Viene con características de alto nivel para facilitar el trabajo de los desarrolladores en el desarrollo de aplicaciones Android. Una de las características esenciales son sus dispositivos virtuales.

**Xcode**: es un IDE de uso común para los sistemas operativos macOS. Utilizando este IDE, los desarrolladores pueden desarrollar software o aplicaciones en Mac que pueden utilizarse en iOS, iPadOS, macOS, tvOS y watchOS

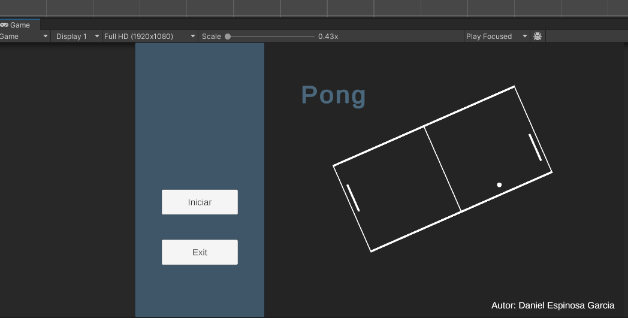


Ilustración 3 Imagen del desarrollo del Pong como primer proyecto de clases

**Unity**: es un motor de videojuego multiplataforma creado por Unity Technologies. Unity está disponible como plataforma de desarrollo para Microsoft Windows, Mac OS, Linux. La plataforma de desarrollo tiene soporte de compilación con diferentes tipos de plataformas entre ellas móviles.

## e) Se han identificado configuraciones que clasifican los dispositivos móviles en base a sus características.

Se podrían clasificar en dos grandes grupos, aunque seguramente se podrían clasificar de muchas otras maneras. Estos dos grupos serían los siguientes:

* **Propósito general**: Incluye dispositivos como computadoras, tablets, relojes y teléfonos inteligentes. Se caracterizan por su capacidad para realizar una amplia gama de actividades y sintetizar funciones. Son el medio de trabajo para muchas personas, permitiendo realizar actividades que generan alta productividad.
  + Teléfonos inteligentes o smartphones. Podemos decir que son como teléfonos móviles convencionales pero construidos sobre una estructura informática móvil, que les proporciona mayor capacidad de almacenamiento y de procesamiento. Se asemejan bastante a un ordenador. Son usados a menudo tanto para el entretenimiento como herramientas de trabajo.
  + Tabletas. Son ordenadores portátiles personales integrados en una pantalla táctil, con acceso a Internet y capacidad para ejecutar aplicaciones instaladas sobre un sistema operativo. Su manejo es fácil e intuitivo.
  + Relojes inteligentes. Son relojes de pulsera que poseen funcionalidades como acceso a Internet, recibir llamadas o enviar mensajes.
  + Ordenadores portátiles. Son ordenadores completos que, debido a su tamaño reducido y compacto, permiten su transporte y gracias a su batería, su uso mientras no esté conectado el cargador a la red eléctrica.
* **Propósito de entretenimiento:** Este grupo incluye dispositivos diseñados para proporcionar diversión a las personas, como aquellos que permiten jugar videojuegos como propósito principal.
  + **Nintendo Switch:** es una consola de videojuegos desarrollada por Nintendo. Fue lanzada mundialmente el 3 de marzo de 2017.
  + **Steam Deck**: es un [PC de juegos](https://es.wikipedia.org/wiki/PC_de_juegos) portátil desarrollado por [Valve Corporación](https://es.wikipedia.org/wiki/Valve_Corporation). Lanzado el 25 de febrero de 2022, Steam Deck se puede usar como PC portátil o de sobremesa usando un monitor externo. El dispositivo está diseñada para soportar gran parte de la biblioteca de Steam utilizando [SteamOS](https://es.wikipedia.org/wiki/SteamOS), un sistema operativo que estriba en un kernel Debian[1](https://es.wikipedia.org/wiki/Steam_Deck#cite_note-1)​. Esto permite el soporte de un gran número de juegos de PC de [Windows](https://es.wikipedia.org/wiki/Windows) a través de la capa de compatibilidad [Proton](https://es.wikipedia.org/wiki/Proton_(software)) utilizando [Linux](https://es.wikipedia.org/wiki/Linux). El sistema es una plataforma abierta, permitiendo a los usuarios instalar otros juegos compatibles con el dispositivo o incluso otros sistemas operativos.

## g) Se ha analizado la estructura de aplicaciones existentes para dispositivos móviles identificando las clases utilizadas.

Para el desarrollo del pong utilizo 6 Scripts.

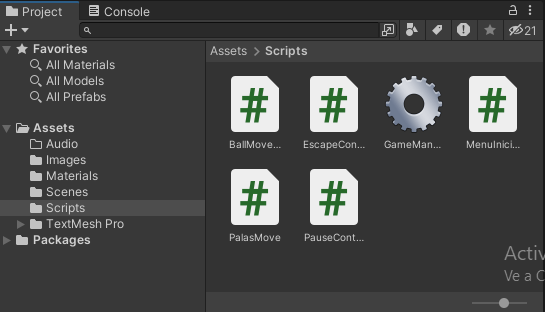


Ilustración 4 Scripst utilizados en el Pong

**Script GameManager** que es el que se encargara de gestionar todo el desarrollo del juego desde el inicio. Utilizando el resto de Los scripts para su funcionamiento.

Controla las puntuaciones de los goles. y el tiempo de espera para relanzar la bola después de que se marca el gol.

**Script MenuInicio** para gestionar el menú de Inicio. El cual se encargará de llamar a la Escena del Juego y También permitirá salir / cerrar el juego.

El resto de Los scripts se encargan del funcionamiento interno del Juego.

**Script PauseControl** que se ocupa de detener y reanudar el juego utilizando la función Time.timeScale que detiene el paso de tiempo en el Juego deteniendo así la bola y las palas.

**Script EscapeControl** permite salir de la escena del juego a la escena del menú principal.

**Script PalasMove** Da funcionalidad a las palas para que se puedan mover cada una independientemente hacia arriba y abajo, teniendo un límite de posición.

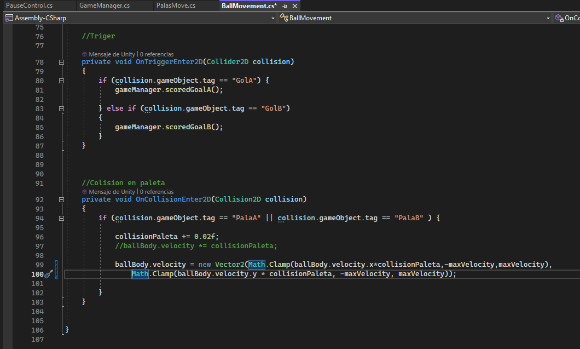


Ilustración 5 Script BallMovement

**Script BallMovement** este además del GameManager es de los más complejos ya que controla los coliders con las palas aumentando la velocidad tanto de la bola como las palas con cada impacto a estas, limitando su velocidad. Esta monitorizando si el triger de las porterías es activado para que el GameManager aumente el marcador y relance la bola desde el centro utilizando un lanzamiento aleatorio.

## h) Se han realizado modificaciones sobre aplicaciones existentes.

Para realizar el Pong no me apoye en la documentación entregada en Moodle, lo realice con la ayuda de compañeros para entender el funcionamiento de Unity, como se conectan los Scripts con los objetos y como se controla el juego desde el GameManager.

Despues de revisar un poco la documentación entregada veo que tengo modificado las siguientes cosas:

* Una escena de menú de inicio que me parece que no está incluida en el documento y añado el botón de Iniciar y el botón de Exit para salir del juego.
* La bola del juego parte con una velocidad inicial y va acelerando a medida que golpea con las palas, añadiendo un límite a la velocidad que puede alcanzar.

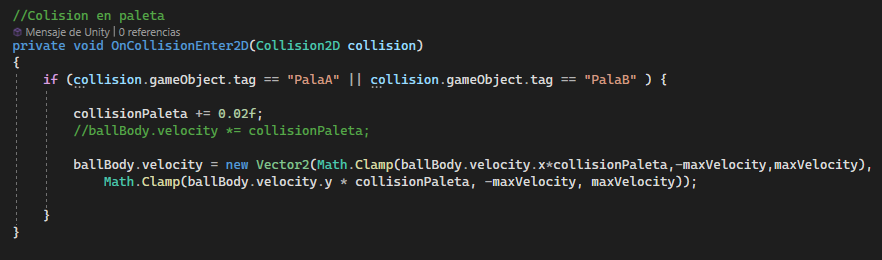


Ilustración 6Script para controlar la aceleración de la bola.

Tampoco invertí mucho más tiempo en este proyecto ya que decidimos entre varios compañeros realizar un proyecto más complejo y grande entre varios.

i) Se han utilizado emuladores para comprobar el funcionamiento de las aplicaciones.

Explicar emuladores mercado

Switch yuzu.

BlueStack.

Emulador móvil unity (hacer foto)

**RA2**

## a) Se ha generado la estructura de clases necesaria para la aplicación.

En el siguiente esquema se muestra el funcionamiento de los Scripts del Juego Pong y como se relacionan entre ellos desde el Inicio del Programa hasta su cierre.

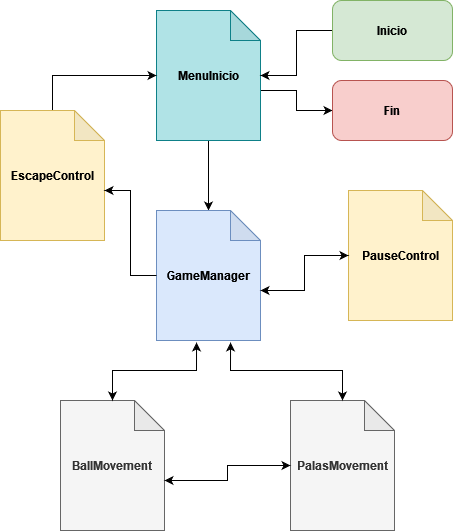


Ilustración 7 Diagrama de como utiliza el Juego Pong los Scripts en la ejecución de la aplicación

El funcionamiento de cada Script ya se realizó en un apartado anterior. RA1 punto g.

## b) Se han analizado y utilizado las clases que modelan ventanas, menús, alertas y controles para el desarrollo de aplicaciones gráficas sencillas.

Para realizar el menú de inicio del juego incorpore otra escena en la cual se añade un canvas para añadir a este los Botones de Iniciar y Exit. Los cuales cambian de color al presionarlos.

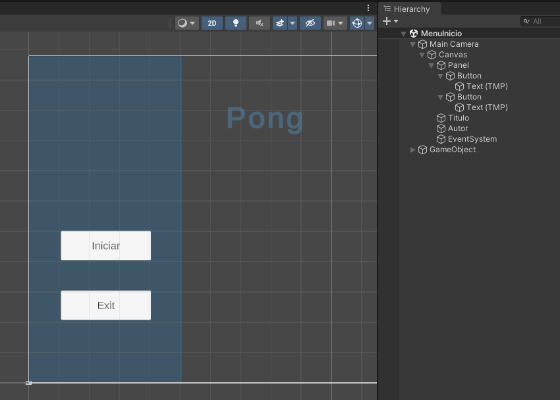


Ilustración 8 Canvas del Menu de Inicio del juego Pong

Se añade dos cuadros de texto para incorporar el Titulo del Juego y el autor de este.

Este utiliza el Script MenuInicio para lanzar la escena del juego.

c) Se han utilizado las clases necesarias para la conexión y comunicación con dispositivos inalámbricos.

investigar

d) Se han utilizado las clases necesarias para el intercambio de mensajes de texto y multimedia.

Albano tambien la tiene Pendiente investigar

## e) Se han utilizado las clases necesarias para establecer conexiones y comunicaciones HTTP y HTTPS.

Revisando la documentación de Unity encontramos la Clase “UnityEngine.Networking” la cual permite gestionar las conexiones y comunicaciones HTTP y HTTPS.

Esta clase implementa metodos para el acceso y el envió de información. Ademas de propiedades para la configuración de la conexión.

Tenemos por ejemplo el método UnityWebRequest.Post que permite enviar información de un formulario a un servidor http.

En el siguiente link tenemos acceso a toda la información de la Clase UnityEngine.Networking

<https://docs.unity3d.com/ScriptReference/Networking.UnityWebRequest.html>

## f) Se han utilizado las clases necesarias para establecer conexiones con almacenes de datos garantizando la persistencia.

Unity no tiene como tal una clase específica para gestionar las conexiones a BBDD, por ende, es necesario apoyarnos en C# .NET, utilizando la clase Microsoft.Data.SqlClient que permite crear la configuración necesaria para conectarnos a una BBDD y trabajar con ella.

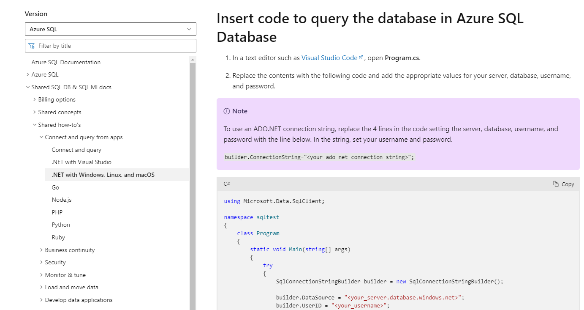


Ilustración 9 Pagina Web Learn.Microsoft.com con la explicacion del codigo para conectarnos a BBDD

De manera local sí que podemos guardar información mediante Unity, podemos utilizar la clase PlayerPref que nos permite almacenar las preferencias del jugador entre las sesiones del juego. Pero esta información no está cifrada por lo cual se puede modificar fácilmente.

Otra forma de poder hacerlo de manera local sería creando desde C# flujos para escribir archivos serializados y guardar la información.

## g) Se han realizado pruebas de interacción usuario-aplicación para optimizar las aplicaciones desarrolladas a partir de emuladores.

A medida que fui desarrollando el Juego fui probándolo para realizar ajustes e implementar funcionalidad y comprobar que todo funcionaba correctamente.

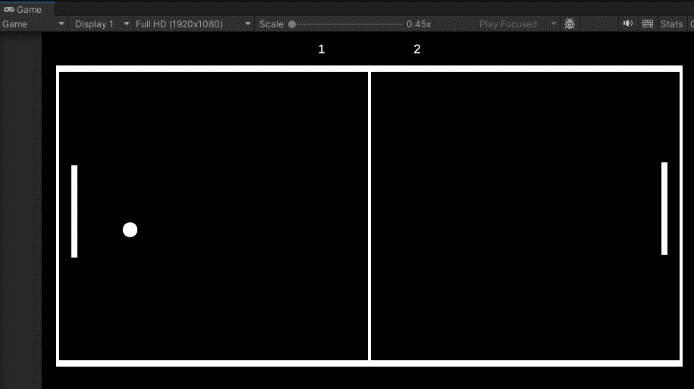


Ilustración 10 Pruebas del Pong utilizando el Emulador de Unity

* Se comprobaron que los marcadores funcionan correctamente.
* Que la bola rebota en el entorno y las palas.
* El pulsado de la barra espaciadora Pausa el juego
* El pulsado de Botón de Escape te devuelve a la escena principal
* Se ajustó la velocidad de inicio de la bola ya que era demasiado baja.
* Se ajustó la velocidad máxima de la bola ya que aceleraba demasiado.
* Se añadió una pausa de 1 segundo despues de marcar un gol para volver a lanzar la bola.

## h) Se han empaquetado y desplegado las aplicaciones desarrolladas en dispositivos móviles reales.

Despues de realizar los ajustes necesarios para poder crear el build instalo el juego en el dispositivo y compruebo que funciona. Me quedo pendiente ajustar el canvas y modificar los controles.

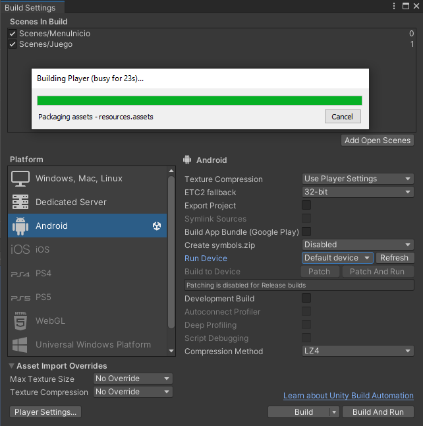


Ilustración 11 Build de APK



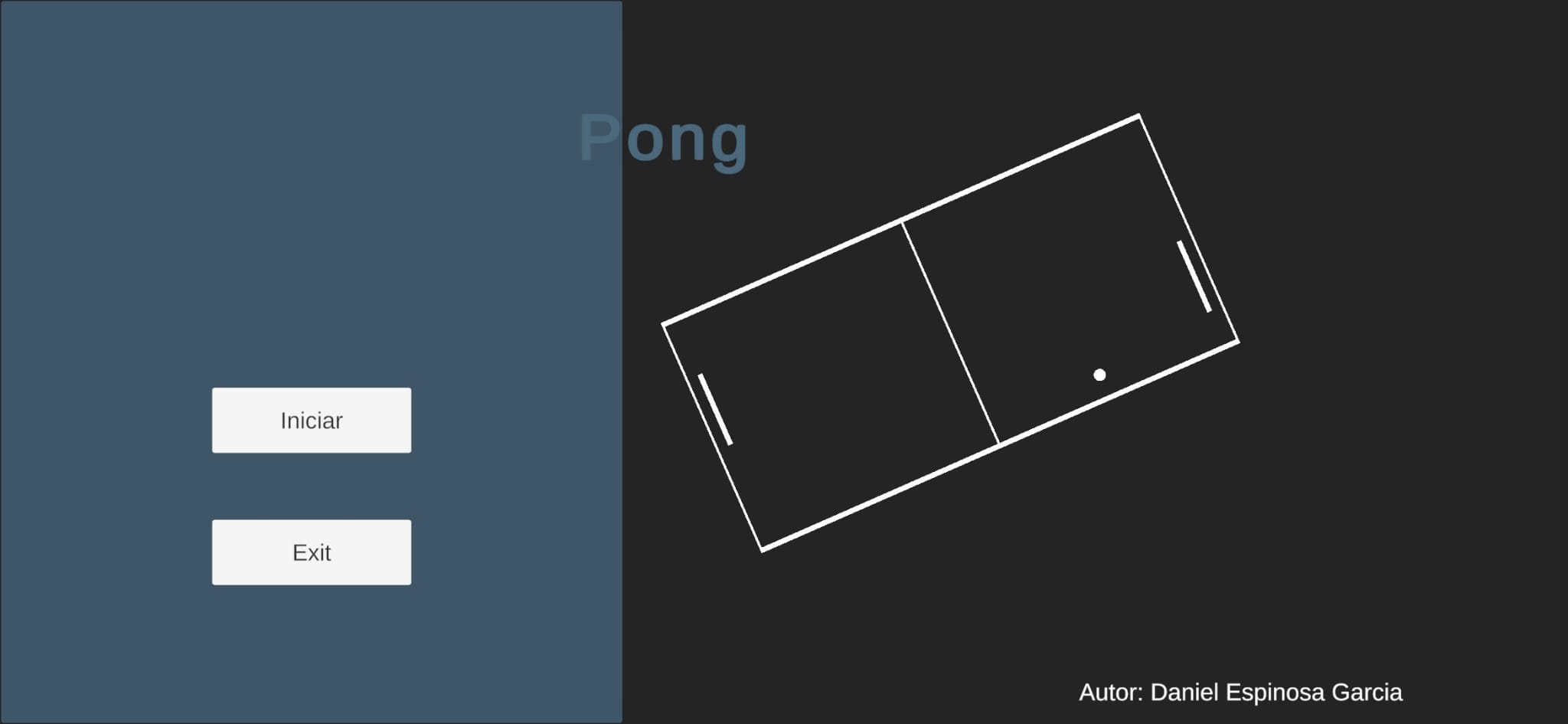
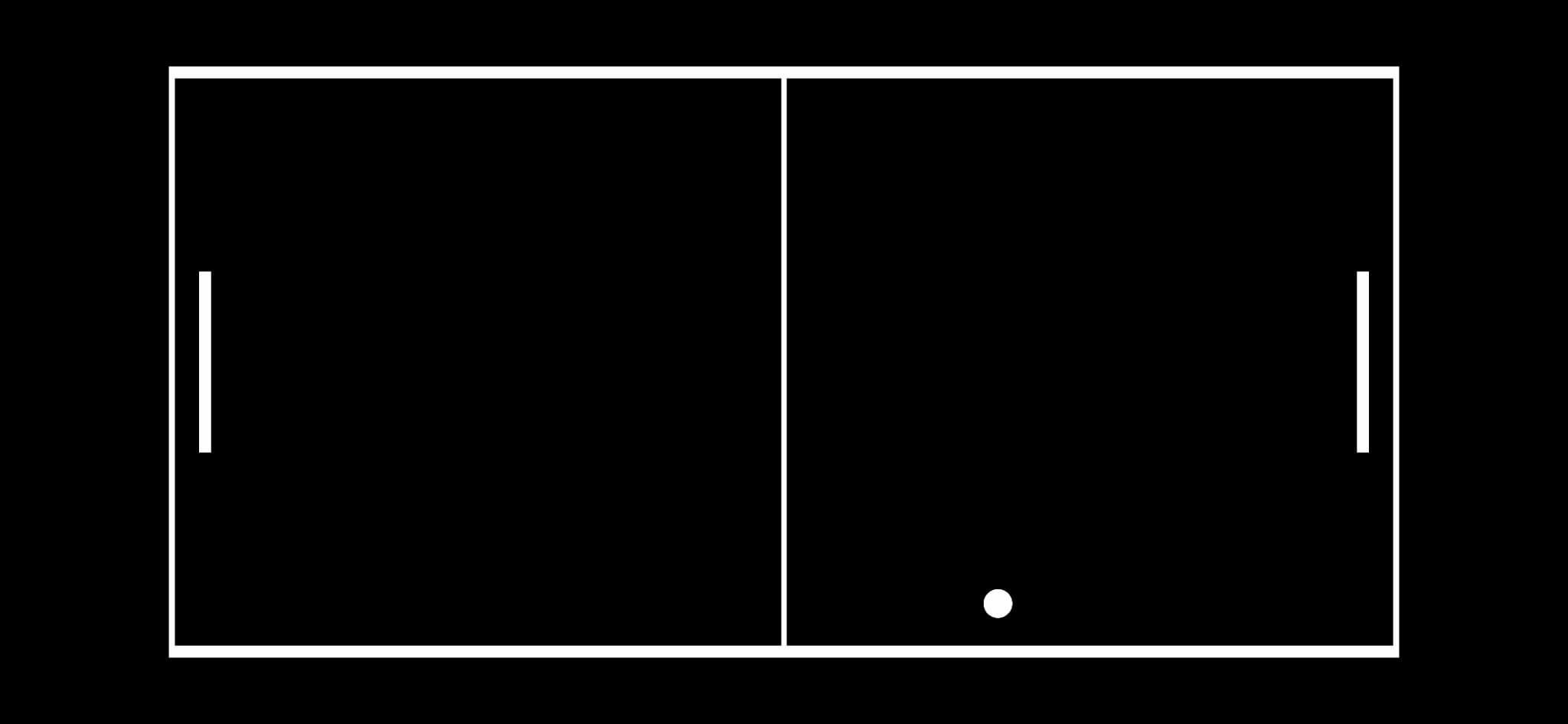
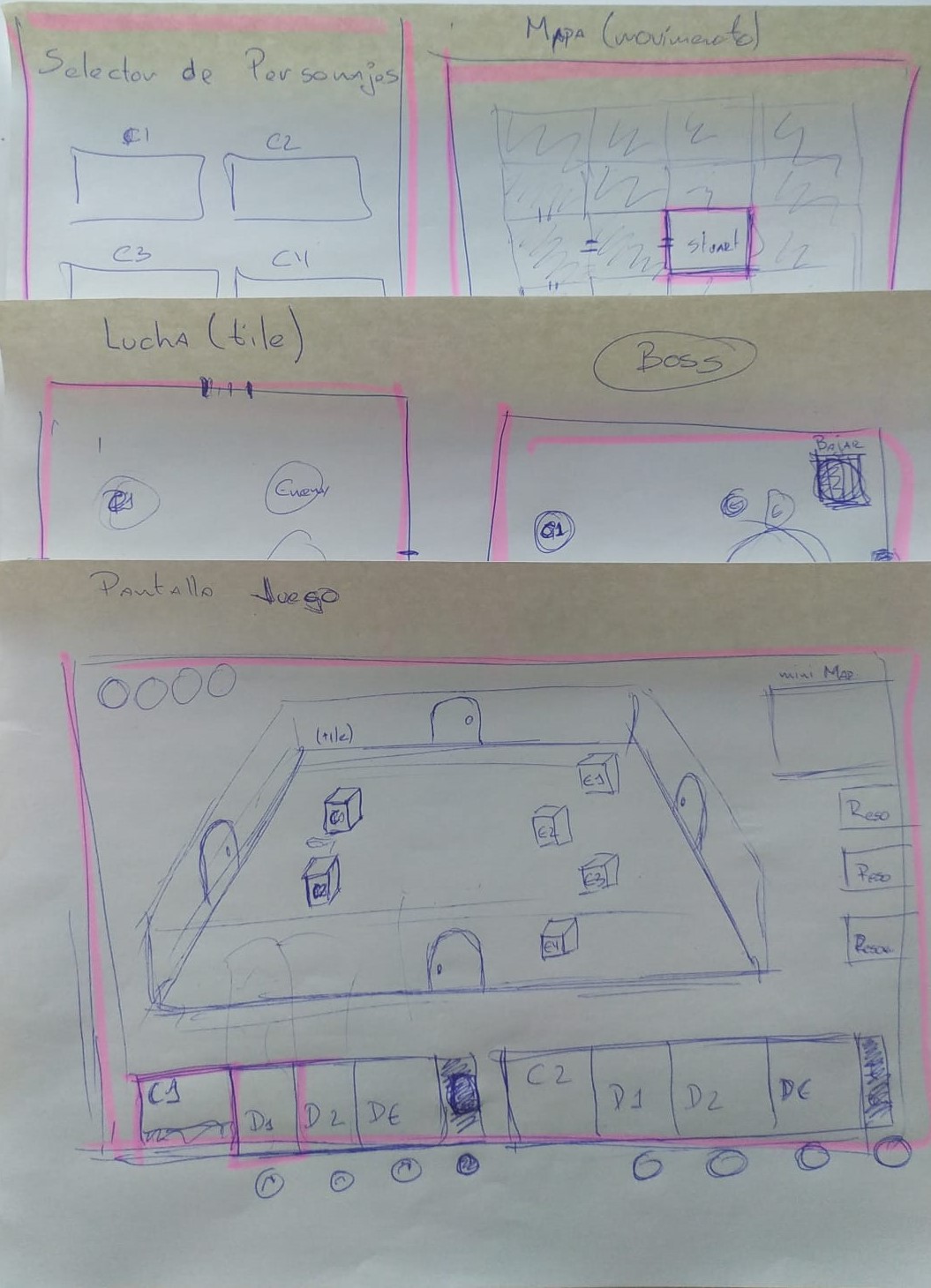


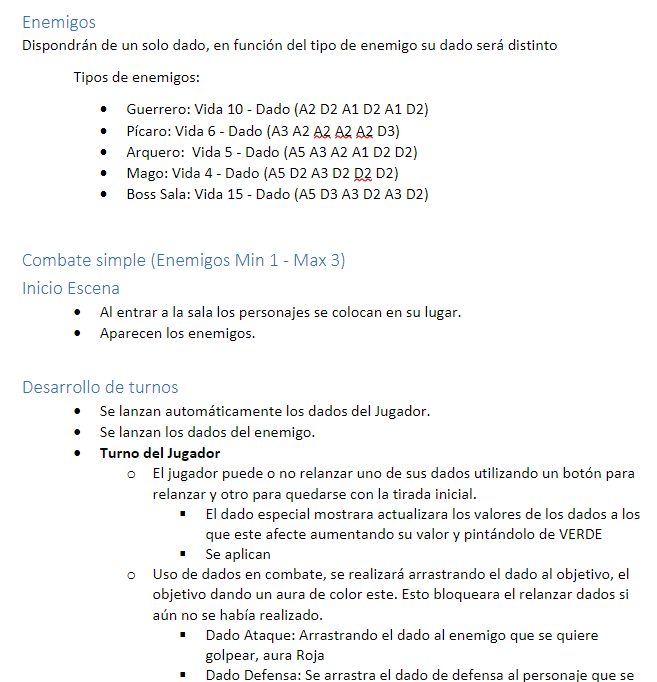
Ilustración 12 Instalación y prueba del Pong en Android

## i) Se han documentado los procesos necesarios para el desarrollo de las aplicaciones.



Para el desarrollo del siguiente juego hemos empezado con la creación del GDD en el cual tenemos algunos dibujos sencillos de cómo se desarrollaran las escenas del juego asi como un documento de texto en el cual se explican las mecánicas del mismo y el concepto general del juego.

En este caso realizaremos un Rogue-Like en el cual dados de 6 caras representaran las armas que utiliza el personaje y sus habilidades de clase a medida que avanza por el mapa generado proceduralmente y enfrenta enemigos, al derrotarlos obtendrá oro el cual podrá emplear para mejorar los dados. Al morir se calculará la experiencia obtenida y esto desbloqueará nuevos dados con los cuales volver a empezar la partida.



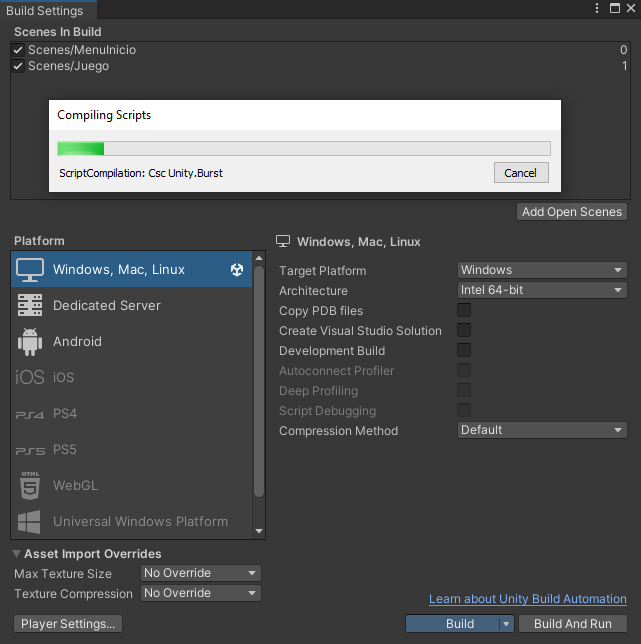


Ilustración 13 Build del Juego Pong para Windows