TRABAJO FIN DE GRADO

GRADO EN DESARROLLO Y DISEÑO DE VIDEOJUEGOS

CURSO ACADÉMICO 2022/2023

CONVOCATORIA JUNIO

Prim[AR]yCode. Aprende Programación en Scratch con Realidad Aumentada

AUTOR: Medina Martínez, José Antonio

TUTORA: Raquel Belén Hijón Neira

En Móstoles a 22 de Junio de 2023

Tabla de contenido

Tabla de imágenes

RESUMEN

En este proyecto se ha desarrollado un videojuego educativo en Realidad Aumentada para móviles, haciendo uso del motor Unity y la API Vuforia destinado a niños de 10 a 12 años.

La aplicación se llama Prim[AR]yCode, remarcando su conexión con la aplicación en la que está basada, PrimaryCode.

Es un juego educativo que busca enseñar y motivar a los jugadores en los temas básicos y principales de la programación (entrada-salida, condicionales, bucles, funciones, arrays, ficheros y recursividad), usando de base el lenguaje de programación Scratch.

En este juego el principal objetivo es resolver puzles usando marcas de realidad aumentada como piezas, teniendo que ser introducidas en un orden específico.

El jugador tendrá que hacer uso de la cámara del móvil para resolver los puzles, y usar las piezas disponibles para ser escaneadas en el juego. Si el orden de las piezas no es correcto o si faltan algunas se mostrarán mensajes de advertencia para hacer las debidas correcciones y se sugieren pistas.

Se tendrá acceso a dos menús sencillos, uno donde el jugador podrá llegar a la selección de nivel, leer las instrucciones y alterar ciertas opciones; y otro de selección de nivel donde podrá elegir el nivel a jugar.

Además, se contará con una web para tener los diseños de las piezas disponibles y usables, así como una introducción e información adecuada para poder empezar a jugar.

Los diseños tienen un formato que incluye información de la marca en los bordes dejando el centro de la pieza libre para añadir la parte visual de la pieza del puzle.

El uso de la realidad aumentada se ha implementado en el motor de Videojuegos Unity haciendo uso de Vuforia, siendo implementada con su base de datos, y usando el lenguaje de programación C#.

La página web se encuentra disponible en el siguiente enlace:

https://sites.google.com/view/prim-ar-y-code/home

SUMMARY

In this project, an educational augmented reality mobile video game has been developed using the Unity engine and the Vuforia API, targeting children aged 10 to 12.

The application is called Prim[AR]yCode, emphasizing its connection with the app it is based on, PrimaryCode.

It is an educational game that aims to teach and motivate players in basic programming concepts (input-output, conditionals, loops, functions, arrays, files, functions and recursion), using the Scratch programming language as a foundation.

In this game, the main objective is to solve puzzles using augmented reality markers as pieces, which need to be placed in a specific order.

The player will need to use the mobile camera to solve the puzzles and scan the available pieces within the game. If the order of the pieces is incorrect or if some are missing, warning messages will be displayed to make the necessary corrections and provide hints.

There will be access to two simple menus: one where the player can access the level selection, read instructions, and adjust certain options, and another for level selection where they can choose the level to play.

Additionally, there will be a website available to provide the pieces available in an usable manner, as well as an introduction and appropriate information to start playing.

The designs have a format that includes marker information on the edges, leaving the center of the piece free to add the visual part of the puzzle piece.

The use of augmented reality has been implemented in the Unity game engine using Vuforia, integrated with its database and programmed in the C# programming language.

The website can be accessed at the following link:

https://sites.google.com/view/prim-ar-y-code/home

1. INTRODUCCIÓN

1.1. Descripción

El objetivo principal del proyecto es desarrollar un videojuego educativo para móviles haciendo uso de la realidad aumentada como mecánica primaria para enseñar a niños de 10 a 12 años sobre estructuras de programación, particularmente en el lenguaje Scratch[1].

Es una extensión a la ya existente aplicación PrimaryCode[2], con la intención de ser complementario a esta, con los mismos conceptos y ejercicios, pero con una implementación y mecánica de enseñanza diferentes con el objetivo de plantear otras maneras de enseñar los conceptos y de tener una herramienta de revisión para confirmar que los conceptos han sido correctamente aprendidos.

En el videojuego, la misión de los jugadores es completar puzles haciendo uso de unas marcas, al estilo QR, que al ser registradas por la cámara del móvil en la aplicación se convierten en piezas de un rompecabezas. El jugador debe ordenar las piezas de los puzles para resolverlos y obtener un feedback correspondiente al nivel.

En la página web compañera de la aplicación, se tendrá acceso en diferentes paginas a los diseños de las marcas que podrán ser registradas en realidad aumentada, además de una introducción y explicación, así como links a la aplicación Primary Code y al enlace de descarga

Se tiene como público objetivo a niños entre 10 a 12 años bastante curiosos con la tecnología y los videojuegos, buscando despertar su interés por el lenguaje de programación y demostrar que es lo que puede hacerse con la realidad aumentada. Buscando una estética simple y vistosa, como puede verse en el menú (Ver Ilustración 1)

***Ilustración 1**: Pantalla de juego*

[1] **https://scratch.mit.edu/**
[2] **https://sites.google.com/view/primarycode-v3/inicio**

1.2. Motivación

La realidad aumentada es una tecnología en constante crecimiento que no solo se aplica en videojuegos con fines de entretenimiento, sino también como una herramienta poderosa para la educación. Al integrar la realidad aumentada en un videojuego atractivo y divertido, se puede despertar el interés del jugador y fomentar el descubrimiento continuo, lo que a su vez lo llevará a cuestionarse cómo funciona y a realizar actividades prácticas.

A medida que se superan los desafíos que se presentan, el estudiante se siente motivado a seguir avanzando y completar correctamente cada reto. La aplicación combina la diversión propia de un videojuego con la oportunidad de adquirir nuevos conocimientos y habilidades en un entorno educativo interactivo.

En el mercado de los videojuegos, ya es común utilizar la realidad aumentada y la realidad virtual para entretenerse de forma autónoma. Sin embargo, si aplicamos estas tecnologías en el ámbito educativo, el sistema de aprendizaje ofrecerá una mayor interactividad y elementos que ayudarán a agudizar ciertos sentidos y capacidades.

2. ESTADO DEL ARTE

En este apartado se describen los diversos recursos disponibles para la realización del proyecto.

2.1. Juegos de puzles.

Se piensa habitualmente que los puzles solo sirven como hobbies o una manera de relajarnos y dejar pasar el tiempo, aunque también tienen múltiples beneficios, uno de ellos ayudar a los niños a desarrollar sus habilidades, la memoria y su estado de ánimo. En los videojuegos se implementa mucho este sistema, ya sea resolver un acertijo, un rompecabezas o memorizar una secuencia para retar al jugador a hacer algo distinto.

Los juegos de puzles han sido populares durante décadas y han evolucionado considerablemente en términos de concepto y mecánicas de juego. Podemos seguir varias tendencias y sus desarrollos recientes en el mundo de los juegos de puzles.

2.1.1. Tendencias.

Puzles basados en física:

Los juegos de puzles que se basan en las leyes de la física se han vuelto cada vez más populares. Estos juegos desafían a los jugadores a resolver puzles utilizando principios como la gravedad, el momento físico y la fricción. Ejemplos destacados incluyen "Portal"[1] y "The Talos Principle"[2].

Puzles narrativos.

Muchos juegos de puzles han adoptado una narrativa más profunda, ofreciendo una experiencia más envolvente. Estos juegos combinan puzles desafiantes con una historia intrigante y a menudo presentan personajes memorables y diálogos significativos. Un ejemplo destacado es "Inside"[3], que combina una jugabilidad de puzles única con una narrativa inquietante.

Juegos de puzles para móviles.

Los dispositivos móviles han impulsado la popularidad de los juegos de puzles gracias a su accesibilidad y facilidad de uso. Los juegos de puzles para móviles suelen ser simples pero adictivos, y se han convertido en una forma popular de entretenimiento en dispositivos móviles. Ejemplos notables incluyen "Candy Crush Saga"[4] y "Monument Valley"[5].

2.1.2. Desarrollos recientes:

Inside (2016)

Inside es un videojuego de plataformas y puzles desarrollado por Playdead, el mismo estudio que creó el aclamado juego "Limbo". El juego se caracteriza por su estilo visual único y su atmósfera opresiva y misteriosa. [3]

En Inside los jugadores asumen el papel de un niño sin nombre que se encuentra en un mundo sombrío y distópico. La jugabilidad se desarrolla en un entorno 2.5D, donde el jugador controla al niño y debe superar una serie de desafíos y obstáculos para avanzar en la historia.

La mecánica principal de juego en Inside consiste en correr, saltar y resolver puzles para abrirse camino a través del entorno. A medida que el jugador avanza, se enfrentará a situaciones cada vez más peligrosas y a desafíos que requieren habilidad y astucia para superar.

Además de las habilidades básicas de movimiento, el jugador también debe interactuar con objetos del entorno y manipular elementos para progresar. Esto incluye acciones como arrastrar cajas, activar interruptores y utilizar otros personajes o criaturas para resolver puzles específicos.

Una característica distintiva de Inside (ver Ilustración 2) es su enfoque en la narrativa ambiental y la presentación visual. La historia del juego se desarrolla sin palabras, a través de la exploración y la interpretación de los eventos que ocurren en el entorno.

Ilustración 2: *Videojuego Inside*

Candy Crush (2012)

Candy Crush (King Digital Entertainment) es un juego para móviles donde el objetivo principal es ordenar los dulces para formar filas y columnas de 3 o más del mismo color, ocasionando que estas desaparezcan y obtener más puntos con el menor número de movimientos posible. [4]

El juego se controla usando solo los dedos y tocando la pantalla, el jugador debe mover los dulces (ver Ilustración 3) arrastrándolos con otros para intercambiarlos de lugar, pensando muy bien en la estrategia que se debe usar para conseguir múltiples puntos cambiando los dulces correctos.

Ilustración 3: *Videojuego Candy Crush*

The Room (2012)

The Room (Fireproof Games) es un juego diseñado principalmente para móviles, es un rompecabezas tridimensional donde el jugador debe resolver diversos acertijos y puzles para ir progresando hasta desvelar el misterio que se oculta detrás. [6]

Al ser un videojuego tridimensional, el jugador puede estudiar todo el modelo y la estructura de la caja, buscando pistas y objetos que le ayuden a resolver otros acertijos que esta contiene. (Ver Ilustración 4) Además, se cuenta con un sistema de inventario que permite conseguir objetos para usarlos en ocasiones futuras.

El puzle se puede resolver en cualquier orden, ocasionando que el jugador deba memorizar las ocasiones en las que le faltaba un objeto para continuar.

El juego se controla con la pantalla táctil, arrastrando el dedo encima del teléfono se puede explorar toda la caja, y pulsando sobre esta se pueden presionar botones y revelar secretos, además, se puede arrastrar el dedo encima en diversos sentidos para ocasionar reacciones distintas en la caja.

También se cuenta con un sistema de cronometro para conocer cuánto tiempo tardó el jugador en resolver el puzle, incitándolo a que lo haga en menos tiempo.

***Ilustración 4**: Videojuego The Room*

Very Little Nightmares (2019)

En Very Little Nightmares[7] (ALIKE Studios y Bandai Namco Entertainment [8]) el jugador toma el control de un pequeño personaje que debe resolver diversos acertijos y puzles para continuar con su travesía.

El jugador toca la pantalla para hacer que su personaje interactúe con el entorno y se mueva, ya que el fondo está mezclado con los objetos interactivos, debiéndose prestar especial atención a los objetos que destaquen o sean más útiles dependiendo del contexto de la situación.

En ciertas ocasiones, aparecerán enemigos de los cuales el jugador debe escapar o el juego se termina si es atrapado por alguno de estos individuos, no hay forma de defenderse, solo ocultarse y tratar de idear alguna forma de escapar de los obstáculos.

La vista del juego en este caso es isométrica combinado con gráficos 2D (Ilustración 5), lo que presenta un escenario con dos puntos de fuga, simulando que todo está en un entorno tridimensional.

Ilustración 5: *Videojuego Very Little Nightmares*

Portal (2017)

Portal es un videojuego de puzles en primera persona desarrollado por Valve Corporation. El juego combina elementos de acción, resolución de puzles y una narrativa intrigante. [1]

La jugabilidad de Portal se basa en el uso de un dispositivo llamado "Portal Gun" que permite a los jugadores crear portales conectados en superficies planas. Estos portales se utilizan para navegar por los niveles del juego, resolver puzles y superar obstáculos.

A medida que el jugador avanza a través de los distintos desafíos, se introduce en una serie de habitaciones de pruebas cada vez más complejas (Ilustración 6). Los jugadores deben utilizar su ingenio y habilidades para manipular el entorno y superar obstáculos, como torretas de seguridad, campos de energía y abismos, utilizando los portales de manera estratégica.

La jugabilidad se centra en la resolución de puzles basados en la física y la lógica espacial. Los jugadores deben aprovechar las propiedades de los portales, como la conservación del impulso y la inercia, para superar los desafíos. A medida que avanzan, se introducen elementos adicionales, como cubos que se pueden llevar y activadores que abren puertas o activan mecanismos.

Ilustración 6: *Videojuego Portal*

2.2. La Realidad Aumentada en los Videojuegos

Muchos juegos para dispositivos móviles usan la realidad aumentada aprovechando el uso de la cámara que la mayoría de estos dispositivos poseen, para simular que el juego realmente se sienta real e interactivo. Usualmente los controles de un videojuego en realidad aumentada son simples, siendo estos controles táctiles o consistentes en la interacción con un objeto virtual moviendo la cámara en tiempo real. También se usan figuras simples u objetos con formas geométricas para que la aplicación pueda identificarlo de una manera mucho más sencilla, la aplicación puede generar varios modelos o figuras dependiendo del objeto que este apuntando la cámara. [9]

La realidad aumentada (AR) ha sido un área emocionante en el desarrollo de videojuegos en los últimos años. Aquí se presentan algunas tendencias y avances en el uso de la realidad aumentada en los videojuegos.

2.2.1. Tendencias.

Juegos móviles basados en AR.

Los juegos móviles han adoptado ampliamente la realidad aumentada, permitiendo a los jugadores interactuar con el mundo virtual en su entorno físico. Pokemon Go [10] es un ejemplo destacado, ya que utiliza la realidad aumentada para permitir a los jugadores capturar Pokémon en el mundo real. Otro similar es Minecraft Earth.[11]

Experiencias inmersivas.

Los desarrolladores están creando experiencias más inmersivas utilizando la realidad aumentada en plataformas como gafas inteligentes y dispositivos de realidad virtual (VR). Estos juegos combinan elementos virtuales con el entorno físico del jugador, brindando una experiencia más envolvente y realista.

Integración de la realidad aumentada en consolas de videojuegos.

Las consolas de videojuegos como la PlayStation 5 y la Xbox Series X han comenzado a incorporar características de realidad aumentada. Esto permite a los jugadores interactuar con objetos virtuales en su entorno físico utilizando cámaras y sensores especiales.

Juegos de mesa y cartas con AR.

La realidad aumentada también se está utilizando para mejorar los juegos de mesa y las experiencias de cartas. Algunos juegos utilizan una aplicación de AR para agregar elementos visuales y efectos especiales a las partidas, brindando una dimensión adicional a la jugabilidad.

2.2.2. Desarrollos recientes:

Pokémon GO (2016)

Pokémon Go (Niantic en colaboración con Nintendo y The Pokémon Company) [10] es un juego de realidad aumentada que utiliza muy bien las mecánicas que esta posee. El objetivo principal del juego es atrapar pokémons, para entrenarlos y hacer que luchen contra otros. El juego aprovecha bastante el uso de detección de planos para generar un modelo que

se adapte al mundo real, y también los controles de la pantalla táctil para realizar las capturas de los pokémons (Ilustración 7).

Ilustración 7: *Videojuego Pokemon Go*

Minecraft Earth (2019)

Minecraft Earth (Mojang Studios) Iilustración 8) fue un ambicioso proyecto en el cual se buscaba combinar dos conceptos: un juego de supervivencia y un juego en realidad aumentada. [11] El objetivo del juego era sobrevivir consiguiendo recursos y enfrentando a monstruos, la forma de hacerlo era que el jugador se moviese con su dispositivo móvil a distintas locaciones para conseguir recursos. El juego se controlaba usando la pantalla táctil y la cámara para escanear el entorno, lo que adaptaba los elementos virtuales a locaciones dependiendo de lo que se escaneara. También se encontraba disponible la construcción, lo que agregaba un atractivo más al juego, pues el jugador podía construir prácticamente lo que quisiese teniendo como único límite su imaginación. Todo lo que el jugador construía o destruía se quedaba registrado en el juego, eso quiere decir que si el jugador pasaba por una misma locación podría ver su progreso ya sea si lo había destruido o construido.

Ilustración 8: *Videojuego Minecraft Earth*

2.3. Diferencias entre realidad aumentada y realidad virtual:

La realidad virtual (RV) y la realidad aumentada (RA) son tecnologías inmersivas que ofrecen experiencias interactivas, pero se diferencian en la forma en que los usuarios interactúan con ellas y la forma en que perciben el mundo real.

La realidad virtual crea un entorno completamente digital e inmersivo que aísla al usuario de la realidad física. Para experimentar la RV, los usuarios usan dispositivos como gafas de RV o cascos que cubren completamente sus ojos y, a veces, sus oídos. Estos dispositivos muestran imágenes y videos en 3D y pueden rastrear los movimientos de la cabeza y los controladores de mano para proporcionar una experiencia envolvente. En la RV, los usuarios están completamente inmersos en un mundo virtual simulado y no pueden ver ni interactuar directamente con su entorno físico.

Por otro lado, la realidad aumentada (RA) combina elementos virtuales con el entorno físico real. A través de dispositivos como smartphones, tablets o gafas especiales, los usuarios ven el mundo real a través de una pantalla mientras se superponen imágenes o información digital en tiempo real. Estos elementos virtuales pueden ser gráficos, texto, videos u objetos 3D que parecen existir en el entorno físico. A diferencia de la RV, la RA no sustituye la realidad física, sino que la enriquece con información adicional o interactividad.

En resumen, la principal diferencia entre la realidad virtual y la realidad aumentada radica en la forma en que los usuarios interactúan con el entorno. La RV sumerge al usuario en un entorno completamente virtual, mientras que la RA superpone elementos virtuales en el mundo real. Ambas tecnologías tienen aplicaciones en diferentes campos, como videojuegos, educación, medicina, arquitectura y más, pero su experiencia y forma de interacción son distintas. [12]

2.4. Conclusiones

Entre los videojuegos analizados no se encuentra ninguno que combine la realidad aumentada con el género de puzles, lo cual haría que Prim[AR]yCode sea un concepto innovador al combinar las dos mecánicas, agregando el valor de educativo y autonomía que incluirá la aplicación. Además, ninguno de los videojuegos escanea un objeto con un diseño definido, esto agrega un valor único a la aplicación desarrollada. Los controles combinan el uso de la pantalla táctil con la cámara, teniendo que alinear también en tiempo real las piezas necesarias para ordenar el rompecabezas.

3. HERRAMIENTAS DE DESARROLLO

En este capítulo se mencionan las herramientas utilizadas en el desarrollo de esta aplicación, explicando también cual ha sido su utilidad.

3.1. Hardware

3.1.1. Ordenador

Para el desarrollo de la aplicación se ha utilizado una herramienta fundamental que es el ordenador, el cual contiene las siguientes especificaciones:

- Sistema Operativo: Windows 11 Home
- Procesador: AMD Ryzen 7 6800H 3.2GHz
- Tarjeta Gráfica: NVIDIA RTX 3060M
- Memoria RAM: 16,00 GB

3.1.2. Teléfono móvil

- Xiaomi Poco F3 3g con Android 11

3.2. Software

3.2.1. Unity 2020.3.36f1

La aplicación se ha realizado en el motor para videojuegos de Unity, que se puede usar de forma gratuita. Usa diversas herramientas que ayudan al desarrollo de la aplicación en Realidad Aumentada y puede distribuirla en distintas plataformas. [13]

3.2.2. Visual Studio 2019

En el ámbito de la programación, se ha utilizado este editor de código, gratuito para estudiantes, que además es compatible con Unity. [14]

3.2.3. CorelDraw 2021

Los diseños y arte implementado en la aplicación se han realizado en CorelDraw, pues es uno de los programas más fáciles de utilizar para realizar dibujos sencillos. [15]

3.2.4. Adobe Ilustrator 2021

Para el diseño de los elementos para escanear se ha utilizado Adobe Illustrator, es uno de los mejores programas para diseñar las figuras que se van a detectar en la aplicación y es compatible con Vuforia Vumark Designer. [16]

3.2.5. Vuforia Engine 10.7

Con el API Vuforia se ha implementado la herramienta de Realidad Aumentada con Unity, multiplataforma, con seguimiento robusto y rendimiento de una variedad de hardware. [17]

3.2.6. Google Sites

Google Sites es una plataforma en línea que permite crear y compartir sitios web de forma sencilla. Con una interfaz intuitiva y características de arrastrar y soltar, puedes personalizar la apariencia y agregar contenido fácilmente. [18]

También ofrece integración con otros servicios de Google y permite la colaboración en tiempo real, lo que la convierte en una herramienta versátil y accesible para crear sitios web profesionales.

4. DESCRIPCION INFORMÁTICA

A continuación, se detallan los diversos aspectos técnicos en los que se basa el desarrollo del proyecto.

4.1. OBJETIVOS

Ya que la idea principal está concretada, es necesario definir y describir la serie de objetivos para la aplicación. La meta primordial del proyecto es resolver los puzles usando la Realidad Aumentada y enseñando al jugador acerca de la programación.

Para enfocar el juego en el área de la educación usando la Realidad Aumentada es importante crear una interfaz amigable, intuitiva y simple.

El concepto principal que se trabaja durante el juego es la resolución de puzles: Cuando el jugador inicie el juego, el único y principal objetivo será resolver y ordenar las piezas del puzle, dando pistas en caso de que se haya equivocado o avisando si está resuelto.

4.2. Requisitos funcionales

En la tabla 1 se describen los requisitos funcionales del Juego con Realidad Aumentada para la enseñanza de la programación en Scratch, por otro lado, en la tabla 2 se recogen los requisitos no funcionales

Tabla 1: Requisitos funcionales

Nombre	Descripción
Instalar la aplicación.	El usuario tiene la capacidad de instalar la aplicación de una manera sencilla en su dispositivo móvil.
Iniciar la aplicación.	El usuario tiene la capacidad de iniciar la aplicación por un ejecutable disponible tras la instalación.
Navegación.	El juego permite al usuario pasar a través de diversas pantallas de forma rápida y sencilla.
Empezar.	Esta opción permite al usuario comenzar la partida del juego.
Feedback.	El juego mostrará pistas sobre si el jugador esta realizando las acciones correctas o se equivocó.
Salir.	El juego siempre mostrara una opción para cerrar la aplicación en cualquier momento.
Audio.	El juego tiene la posibilidad de desactivar los efectos de sonido y música en cualquier momento.

4.3. Requisitos no funcionales

Tabla 2: Requisitos no funcionales

Nombre	Descripción
Sistema Operativo.	Android
Interfaz gráfica.	El sistema debe proporcionar una interfaz sencilla de comprender para el usuario.
Usabilidad.	El sistema debe tener una dificultad de uso baja.

4.4. Diseño

En esta sección del documento se explican las decisiones de usabilidad, diseño y se analizan cada una de las pantallas y menús que componen la aplicación. El juego tiene una estética bastante simple con colores llamativos, para que el jugador se centre más en las mismas mecánicas.

4.4.1. Interfaz

El juego tiene una interfaz principal que utiliza colores principalmente vibrantes para llamar la atención y resultar más intuitivo.

También se utilizan diversos y diferentes botones con iconos fácilmente reconocibles para el jugador

4.4.2. Menú de Inicio

Cuando el usuario inicie la aplicación, la primera pantalla que se presentará será el menú de inicio (ilustración 9). En la parte superior central aparece el título del juego, y en la parte inferior central se mostrarán 3 botones: Empezar partida, opciones y salir.

Ilustración 9: *Menu principal*

Empezar partida nos llevará un menú adicional (ilustración 10) de selección de nivel que nos enviará al primer ejercicio del nivel elegido.

Ilustración 10: *Menu de selección de nivel*

4.4.3. Interfaz de Juego

La interfaz de juego es bastante simple y sencilla. Consta principalmente de señales que aparecerán después de que alguno de los puzles esté resuelto.

Dispone de varios botones, para avanzar entre los distintos puzles, mostrar y ocultar el enunciado, y volver al menú de inicio.

Una vez las piezas sean escaneadas, la pista cambiará dependiendo si el orden de las piezas es correcto o no, si las piezas están en el orden correcto se mostrará un feedback que el usuario podrá leer libremente con la cámara siempre y cuando las piezas estén disponibles.

En la parte superior izquierda de la pantalla se encontrará disponible en todo momento un botón para permitir al usuario regresar al menú principal.

4.4.4. Opciones

En el menú de opciones está disponible el cambio de volumen de audio (ilustración 11), tendrá un slider que el jugador puede alterar moviéndolo de izquierda a derecha, y un botón en la parte inferior que permitirá al usuario regresar al menú principal una vez que haya ajustado el volumen a su preferencia.

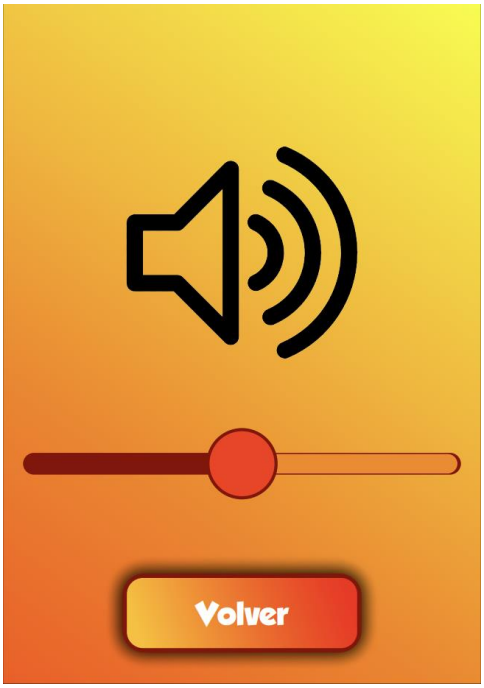

***Ilustración 11**: Opciones (Volumen)*

4.5. IMPLEMENTACIÓN

En este capítulo se van a explicar los aspectos más importantes de la implementación llevada a cabo en la aplicación, como el funcionamiento interno que esta posee, las clases principales y los códigos más relevantes.

La aplicación cuenta principalmente de tres escenas bases: El menú principal, el menú de selección de puzle, y los juegos. Cada puzle por separado conforma una escena diferente.

4.5.1. Código

VumarkCantidad

La clase VumarkCantidad se encarga principalmente de detectar que las piezas se encuentren en el lugar correcto. (Ilustración 12)

Además, existen otras clases que utilizan diversas funciones que se encuentran en el VumarkCantidad, tal y como se describirá detalladamente en los siguientes apartados.

- Volumen: Se encarga de controlar el volumen del juego, pudiendo ser alterado por el jugador en el Menú principal.
- VuMarkHandler: Esta clase se encarga de mostrar la pieza correspondiente al VuMark dependiendo del ID identificado.
- MainMenu: Con esta clase se puede cambiar de escena y salir del juego, principalmente
- PiezasEvent: Esta clase sirve para que la Piezas detectadas tengan efecto según si la pieza detectada es correcta
- Distance: Con esta clase se detecta la distancia entre las piezas y si están en el orden correcto.

Cuando el juego se inicie, en cada Update el VumarkCantidad se encargará de detectar la cantidad de piezas en pantalla, y si están en el orden correcto. (Ilustración 12)

```
if (Nivel == 1)
{
    if (Pieza1 && Pieza2 && Pieza3 && Pieza4)
    {
        Correcto = true;
    }
    if (Puzzles == 4)
    {

        if (!Correcto)
        {
            PiezasIncorrectas.SetActive(true);
            PiezasFaltantes.SetActive(false);
            Resuelto.SetActive(false);
        }
        if (Correcto)
        {
            Resuelto.SetActive(true);
            PiezasIncorrectas.SetActive(false);
            PiezasFaltantes.SetActive(false);
        }
    }
}
```

Ilustración 12: *Función "Update"*

En la función Update dependiendo de la cantidad de Puzles detectados, y si es que están en el orden correcto, se mostraran textos sobre sí faltan Piezas, son las incorrectas, o son correctas y el puzle se ha resuelto.

Si se detectan los Puzles necesarios y se encuentran en el orden correcto, se activará la condicional que mostrará el rompecabezas resuelto. (Ilustración 12)

Una vez que el Puzle se haya resuelto, se procederá a avanzar al siguiente nivel.

Se tienen dos funciones públicas que indican si se ha aumentado o disminuido la cantidad de piezas detectadas actuales, el cual es principalmente invocado por las piezas de rompecabezas una vez que sean detectados por la cámara o se pierdan de vista. (Ilustración 13)

```
public void AumentarPiezas()
{
    Puzzles += 1;
}

public void DisminuirPiezas()
{
    Puzzles -= 1;
}
```

Ilustración 13: *Funciones "AumentarPiezas" y "DisminuirPiezas"*

Se han creado dos funciones, por pieza, que identifican si la Pieza corresponde al nivel y su existencia (Ilustración 14).

```
public void Pieza1BoolTrue()
{
    if (Pieza1 == false)
    {
        Pieza1 = true;
    }
}
public void Pieza1BoolFalse()
{
    if (Pieza1 == true)
    {
        Pieza1 = false;
    }
}
```

Ilustración 14: *Funciones "Pieza1BoolTrue" y "Pieza1BoolFalse"*

Volumen

El Volumen se encargará de mantener el volumen elegido para la música y los efectos de sonido, el jugador debe ingresar al menú de opciones para poder alterarlo a su gusto. En la función Start (ilustración 15), siempre se recuperará el valor guardado del Audio, el cual adaptará la posición del slider según su ubicación.

```
{
    public Slider slider;
    public float sliderValue;
    void Start()
    {
        slider.value = PlayerPrefs.GetFloat("volumenAudio", 0.5f);
        AudioListener.volume = slider.value;
    }
```

Ilustración 15: *Variables de Volumen y la función "Start"*

La función de ChangeSlider() (ver ilustración 16) se encarga de guardar el valor actual del volumen y alterarlo cada vez que la posición del Slider cambie, este se guarda en todas las escenas llamando a PlayerPrefs.

```
public void ChangeSlider(float valor)
{
    sliderValue = valor;
    PlayerPrefs.SetFloat("volumenAudio", sliderValue);
    AudioListener.volume = slider.value;
}
```

Ilustración 16: *Función "ChangeSlider"*

VuMarkHandler

En esta clase (ilustración 17) se mostrarán las piezas que corresponden a cada VuMark en base a su ID.

```
{
    public AugmentationObject[] AugmentationObjects;
    static Dictionary<string, Texture2D> sVuMarkInstanceTextures;
    Dictionary<string, GameObject> mVuMarkAugmentationObjects;

    void Start()
    {
        sVuMarkInstanceTextures = new Dictionary<string, Texture2D>();
        mVuMarkAugmentationObjects = new Dictionary<string, GameObject>();

        foreach (var augmentationObject in AugmentationObjects)
            mVuMarkAugmentationObjects.Add(augmentationObject.VuMarkID, augmentationObject.Augmentation);

        VuforiaApplication.Instance.OnVuforiaStarted += OnVuforiaStarted;
        VuforiaApplication.Instance.OnVuforiaStopped += OnVuforiaStopped;
    }
```

Ilustración 17: *Variables de VuMarkHandler y función "Start"*

Se crea una lista que se puede alterar dependiendo cuantos modelos o Vumarks se utilizaran, cuáles son sus ID's correspondientes y se guarda la información de las texturas para que no se cree una nueva cada cierto tiempo. También se detecta si el modelo se encuentra disponible cuando Vuforia ha iniciado o está en un estado de Pausa

La función de OnVuMarkFound() (ver ilustración 18) es la que muestra el modelo dependiendo del ID del VuMark. Se asegura de detectar si ya existen modelos asociados al VuMark detectado y se encarga de destruirlos si es así antes de mostrar el nuevo modelo.

```
public void OnVuMarkFound(VuMarkBehaviour vuMarkBehaviour)
{
    Debug.Log($"<color=cyan>VuMark: {vuMarkBehaviour.TargetName}, ID:{vuMarkBehaviour.InstanceId} tracked</color>" );
    DestroyChildAugmentations(vuMarkBehaviour);
    var augmentation = GetVuMarkAugmentation(VuMarkUtilities.GetVuMarkId(vuMarkBehaviour));
    if (augmentation != null)
        augmentation.transform.SetParent(vuMarkBehaviour.transform, false);
}
```

Ilustración 18: *Función "OnVumarkFound"*

Luego de identificar el ID, se crea una copia del modelo correspondiente, colocándolo como contenido del mismo VuMark. Además, se mostrará en la consola de Unity el nombre y el ID del VuMark que se detectará por la cámara

La función OnVuMarkLost() (ver ilustración 19) se encargará de detectar si el VuMark ya no se encuentra visible, y en base a eso se destruirá la copia del modelo correspondiente. Esto también se mostrará en la consola de Unity, indicando que VuMark e ID han sido destruidos.

```
public void OnVuMarkLost(VuMarkBehaviour vuMarkBehaviour)
{
    Debug.Log($"<color=cyan>VuMark: {vuMarkBehaviour.TargetName}, ID:{vuMarkBehaviour.InstanceId} lost</color>" );

    DestroyChildAugmentations(vuMarkBehaviour);
}
```

Ilustración 19: *Función "OnVumarkLost"*

La función RetrieveStoredTextureForVuMarkTarget() (ver ilustración 20) se encarga de detectar el ID a base de la textura utilizada. En caso de que no exista una textura detectada en el cache o database, se creara una.

```
public static Texture2D RetrieveStoredTextureForVuMarkTarget(VuMarkBehaviour vuMarkTarget)
{
    Texture2D vuMarkTexture;
    sVuMarkInstanceTextures.TryGetValue( VuMarkUtilities.GetVuMarkId(vuMarkTarget), out vuMarkTexture);
    if (vuMarkTexture == null)
    {
        vuMarkTexture = VuMarkUtilities.GenerateTextureFromVuMarkInstanceImage(vuMarkTarget);
        sVuMarkInstanceTextures.Add(VuMarkUtilities.GetVuMarkId(vuMarkTarget), vuMarkTexture);
    }

    return vuMarkTexture;
}
```

Ilustración 20: *Función "RetrieveStoredTextureForVuMarkTarget"*

La función de GetVuMarkAugmentation() (ver ilustración 21) detecta cual de todos los ID's se ha identificado y es la que escoge cual va a ser el modelo mostrado dependiendo del VuMark. En caso de que no se detecte algún modelo, no se mostrara nada.

```
GameObject GetVuMarkAugmentation(string vuMarkId)
{
    GameObject sourceAugmentation;
    mVuMarkAugmentationObjects.TryGetValue(vuMarkId, out sourceAugmentation);

    if (sourceAugmentation == null)
        return null;

    return Instantiate(sourceAugmentation);
}
```

Ilustración 21: *Función "GetVuMarkAugmentation"*

Las funciones OnVuforiaStarted() y OnVuforiaStopped() (ver ilustración 22) sirven para indicar que cantidad máxima de VuMarks serán detectados en caso de que Vuforia esté funcionando o se haya detenido.

```
void OnVuforiaStarted()
{
    VuforiaBehaviour.Instance.SetMaximumSimultaneousTrackedImages(10);
}

void OnVuforiaStopped()
{
    VuforiaBehaviour.Instance.SetMaximumSimultaneousTrackedImages(4);
}
```

Ilustración 22: *Funciones "OnVuforiaStarted" y "OnVuforiaStopped"*

Main Menu

Esta clase es utilizada principalmente para cambiar de escenas y que el jugador pueda explorar los menús e iniciar el juego.

La función de CambiarEscena() (ver ilustración 21) es la que permite cambiar de escena entre las que existen, principalmente entre el menú y la escena de juego principal del nivel, y entre distintas escenas de juego.

Esta es llamada por los botones de “Iniciar Partida” del Menú Principal, el botón de regresar en el gameplay, y los botones de avanzar y retroceder en la partida también. Se puede cambiar a que escena se desea ir colocando el nombre correspondiente.

La función Salir() sirve para salir de la aplicación, es utilizada por el botón “Salir” ubicado en el Menú Principal.

La función de URL() es para acceder a un enlace de internet, abriéndolo en el explorador principal del dispositivo utilizado. Planteado para usarlo para acceder a la web complementaria.

```
public void CambiarEscena(string scenename)
{
    SceneManager.LoadScene(scenename);
}
public void Salir()
{
    Application.Quit();
}

public void URL()
{
    Application.OpenURL("Url");
}
```

Ilustración 23: *Funciones "CambiarEscena", "Salir" y "URL"*

PiezasEvent

Esta clase se ubica en cada una de las Piezas y se encarga de invocar eventos en cada una de estas dependiendo del nivel actual en el cual se encuentre el jugador. Los eventos invocados serán los que se encarguen de reconocer si la Pieza pertenece al rompecabezas actual o no. También se llama a la clase de VumarkCantidad para utilizar sus funciones

La función de OnBecameVisible() (ver ilustración 24) sirve para detectar cuando la pieza sea visible, cuando esto suceda también se activará la función del VumarkCantidad de AumentarPiezas(), esto hará que se sume una a la cantidad actual de Piezas totales siempre que el modelo sea visible. También se identificará cual es el nivel actual del VumarkCantidad, y dependiendo de cuál sea el número se activará uno de los eventos. Estos diferentes niveles se usan para representar y controlar de cuantas piezas consta un puzle.

```
void OnBecameVisible()
{
    VC.AumentarPiezas();
    if (VC.Nivel == 1)
    {
        Nivel1.Invoke();
    }
}
```

Ilustración 24: *Función "OnBecameVisible"*

La función OnBecameInvisible() (ver ilustración 25) se activará cuando no se detecte la pieza visible en la cámara, lo que ocasionará a su vez a que se llame a la función del VumarkCantidad DisminuirPiezas(), lo que reducirá la cantidad de piezas actuales.

```
void OnBecameInvisible()
{
    VC.DisminuirPiezas();
    if (VC.Nivel == 1)
    {
        Lost1.Invoke();
    }
```

Ilustración 25: *Función "OnBecameInvisible"*

En el caso de que la pieza deje de ser visible y esta haya sido la correcta, la condicional que indicaba que esta era la correcta se desactivara y no podrá ser activada de nuevo hasta que el objeto vuelva a ser visible nuevamente.

También se llamará a la función de Destroy(gameObject) (ver ilustración 26) para destruir el modelo cuando no se encuentre visible, y así evitar que se creen más clones de los necesarios. Esta función solo destruirá el clon y no el modelo original, por lo cual no habrá ningún error si es que otra función no llega a destruir al clon.

```
Destroy(this);
```

Ilustración 26: *Función "Destroy"*

Distance

Con esta clase se detecta la distancia y el orden de las figuras dependiendo del nivel, también es esta la que mostrara el feedback correspondiente al nivel cuando todas las figuras sean detectadas y a una distancia determinada.

En la función Update() se detectará el nivel, para saber la cantidad de piezas, llamando a la clase de VumarkCantidad y detectando si las piezas están visibles y disponibles. Si las piezas son las correctas, se determinará si están en la distancia correspondiente y dependiendo de esta se mostrará el Feedback y se invocará un evento

En este ejemplo se detecta el primer nivel, para dos piezas, y si las piezas que se detectan son las correspondientes. Además, se crean variables que determinan la distancia entre todas las piezas. Si las cumplen con las condiciones del script, se mostrará el botón de continuar (ilustración 27).

```
if (VC.Nivel == 1 && CentroPieza1 != null && CentroPieza2 != null && CentroPieza3 != null && CentroPieza4 != null)
{
    float distance1 = Vector3.Distance(CentroPieza1.transform.position, CentroPieza2.transform.position);
    float distance2 = Vector3.Distance(CentroPieza1.transform.position, CentroPieza3.transform.position);
    float distance3 = Vector3.Distance(CentroPieza3.transform.position, CentroPieza4.transform.position);
    float distance4 = Vector3.Distance(CentroPieza4.transform.position, CentroPieza2.transform.position);
    RompeCabezas1.SetActive(false);
    if (distance1 > 1 && distance2 > 1 && distance3 > 1 && distance4 > 1)
    {
        RompeCabezas1.SetActive(false);
    }
    else if (0.26 > distance1 && 0.26 > distance2 && 0.26 > distance3 && 0.26 > distance4)
    {

        RompeCabezas1.SetActive(true);
        Evento1.Invoke();
    }
}
```

Ilustración 27: *Condicionales de Distance*

4.5.2. Diseño e identificación de piezas

Las piezas se dividen en dos partes:

El centro, que contiene el “contenido” de la pieza relacionado con el puzle en cuestión

El borde, que incluye la información que permite a la aplicación reconocer la aparición o desaparición de la pieza en la cámara, así como su identidad.

Este borde identificativo se ha realizado como un Vumark (ilustración 28), que es una herramienta que proporciona la librería usada Vuforia[3] y es creado mediante Adobe Illustrator [16] en base a unas instrucciones y un script proporcionado que lo adapta.[4]

[3] **https://developer.vuforia.com/**
[4] **https://library.vuforia.com/design-guide/designing-vumark-adobe-illustrator**

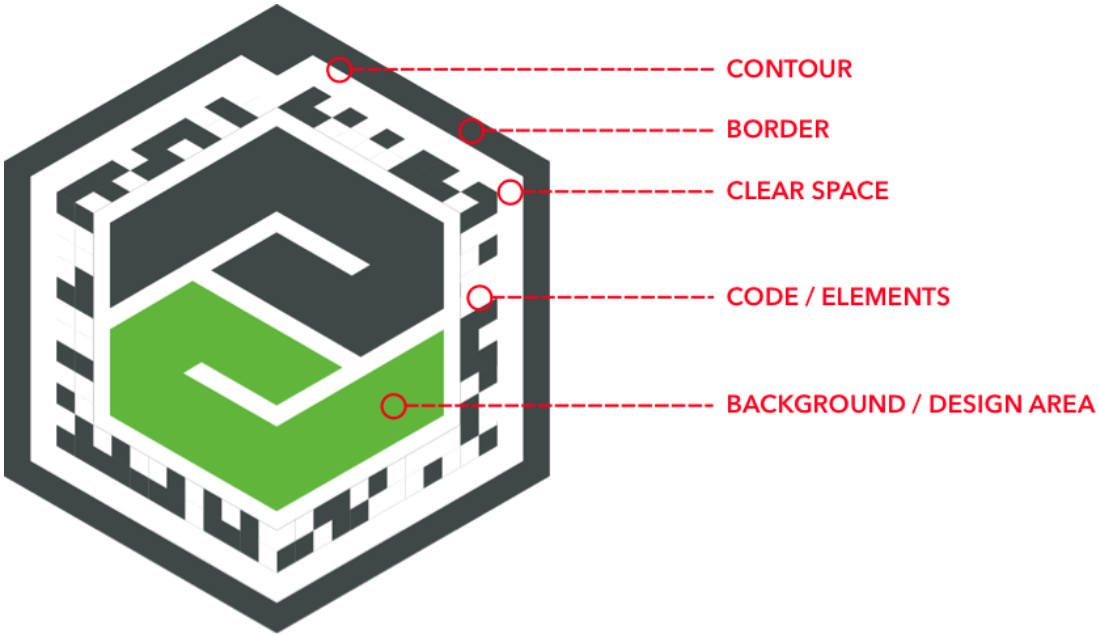

***Ilustración 28**: Secciones de un Vumark ejemplo*

Este Vumark está conformado por un borde plano, que no es simétrico para que la aplicación pueda identificar la orientación de la pieza. Luego dispone de 133 rombos, que según su localización y color (Blanco y negro) proporcionan una información correspondiente a un String de 14 caracteres. La cantidad de elementos identificativos es definida al crear el Vumark (Ilustración 29). Hay que seguir ciertos pasos en base a unas directrices definidas por la plantilla (Ilustración 30). Algunos de los elementos del diseño final son "falsos", es decir, son siempre blancos y están solo para mantener el patrón visualmente.[5]

Finalmente, acabamos con un diseño propio y usable por el motor de Vuforia incorporado en Unity. (Ilustración 31)

[5] **https://library.vuforia.com/vumarks/vumark-design-guide**

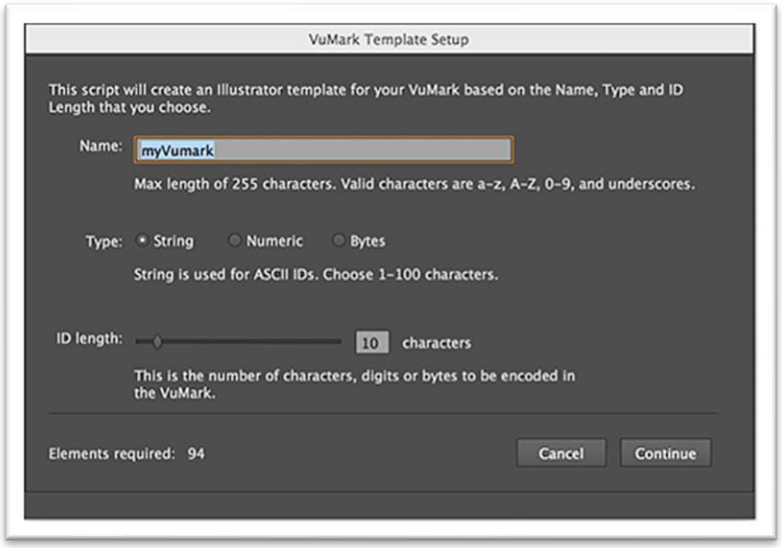

Ilustración 29: *Menu de Script de creación de Vumark*

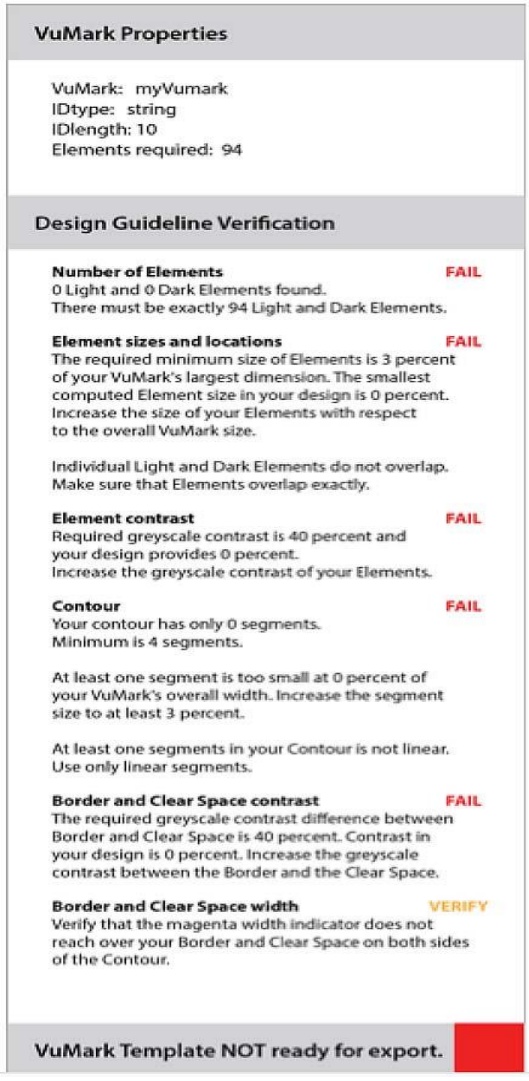

Ilustración 30: *Características y verificaciones de un Vumark*

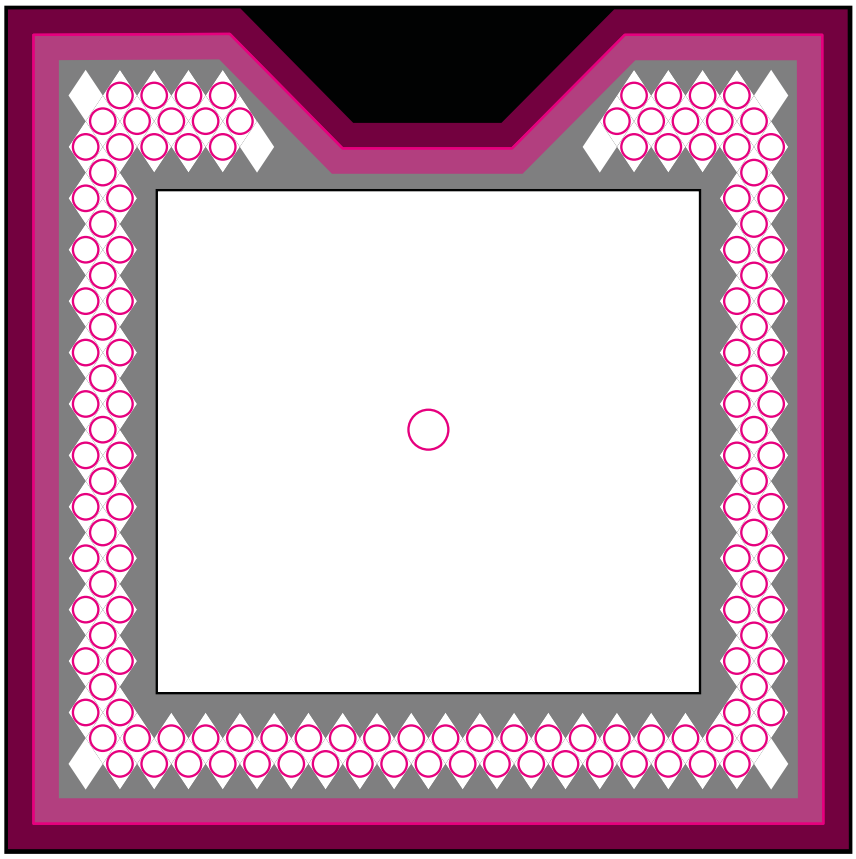

***Ilustración 31**: Resultado creación de mi Vumark*

Este Vumark se sube posteriormente a la página de Vuforia, que te proporciona una clave y un par de archivos .dat y .xml que contienen la información de estos Vumark y que es posteriormente añadido al motor de Vuforia en Unity para que sea capaz de identificarlas.

Aquí se muestra un ejemplo del Vumark representativo del String “Test” (ver ilustración 32).

***Ilustración 32:** Ejemplo de mi Vumark con el String "Test"*

4.5.3. Diseño de niveles

El diseño de los niveles se ha realizado en base a los ejercicios del programa Primary Code, buscando adaptar al modelo de puzles los mismos conceptos enseñados en este.

Los puzles están divididos en un enunciado que explica el problema a resolver, y las piezas que, colocadas las correctas en la posición adecuada, generan visualmente la solución. Estos sets de piezas pueden ser dos que hay que colocar verticalmente de la manera correcta, o cuatro que hay que colocar de manera correcta formando un cuadrado.

Dichas soluciones se dividen principalmente en mostrar con las piezas:

- ✓ El resultado de ejecutar un código mostrado en el enunciado
- ✓ El flujo que resulta de ejecutar un código mostrado en el enunciado
- ✓ El código que realiza lo pedido en el enunciado
- ✓ El código que devuelve el resultado mostrado en el enunciado

Estos puzles, como combinación de piezas y enunciados, han sido sacados directamente de los ejercicios de programación en Scratch de PrimaryCode [0], pero planteándolos de maneras diferentes y adaptados al formato de puzles a resolver e introduciendo el código Scratch de una manera más directa.

De esta manera también si un usuario realizara primero los ejercicios de una de las aplicaciones, podría luego realizar los de la otra y así confirmar que ha aprendido correctamente los conceptos, al ser en su base los mismos ejercicios, pero con distintas implementaciones de realización.

4.5.4. Página web complementaria

Se ha creado también una sencilla página web realizada con Google Sites, que muestra una introducción a la aplicación e instrucciones de uso, así como un enlace para poder descargar el instalable y un enlace a la propia web de la aplicación hermana PrimaryCode (ilustración 33).

El uso principal de esta web es tener una herramienta que permita a los usuarios colocar las piezas para luego poder escanearlas con la aplicación desde el móvil.

Esto se consigue mediante diferentes páginas dentro de la jerarquía de esta web, cada una relacionada con uno de los puzles y compartiendo su título para su fácil localización.

En cada página se mostrará de nuevo el enunciado (ilustración 34), así como varios carruseles de imágenes donde hay que elegir la pieza correcta de entre todas las disponibles a ese ejercicio, es decir, tanto las que irían en otras posiciones como otras que no son acordes al ejercicio sea por mostrar un valor erróneo o un segmento de código que no aporta al resultado.

Página web disponible aquí: **https://sites.google.com/view/prim-ar-y-code/home**

Ilustración 33: *Página principal de la Web complementaria"*

Ilustración 34*: Ejemplo de ejercicio en la web complementaria*

5. CONCLUSIONES

El objetivo principal del proyecto era desarrollar una aplicación que capturara la atención de los usuarios, niños de 10 a 12 años, a través de su principal atractivo, la Realidad Aumentada; y despertar su interés por el campo de la programación, particularmente mediante el lenguaje Scratch y basándose en los ejercicios de la aplicación complementaria PrimaryCode.

En Prim[AR]yCode, se utiliza la cámara del teléfono móvil como herramienta principal para aprovechar la Realidad Aumentada, y mediante el uso de Vumarks que simulan piezas de un rompecabezas, se logra crear un juego de puzles en Realidad Aumentada que ofrece definiciones relacionadas con la programación como recompensa.

El juego presenta una estética simplista con colores llamativos y utiliza menús y botones intuitivos y sencillos. El flujo de la aplicación está formado por el menú principal y el menú de selección de nivel, y las escenas que componen los diferentes niveles de la jugabilidad. En total más de 40 escenas de puzles divididos entre los 7 niveles (Entrada/Salida, condicionales, bucles, arrays, ficheros, funciones y recursividad)

Desde el punto de vista técnico, resulta interesante el logro de detectar múltiples Vumarks con diferentes ID y que cada uno represente una pieza distinta dependiendo del ID detectado. Además, se ha logrado detectar el orden y la distancia entre las piezas, lo que requiere que estén dispuestas en un orden específico para ser reconocidas correctamente.

La aplicación brinda la oportunidad de aprender conceptos básicos de programación como recompensa por resolver los puzles. Podemos afirmar que se ha alcanzado el objetivo principal de desarrollar una aplicación que resulte atractivo al público objetivo y genera interés en los conceptos fundamentales de la programación tanto de manera general como en Scratch.

BIBLIOGRAFÍA

[1] Steam (s.f.). *Portal.* https://store.steampowered.com/app/400/Portal/

[2] Steam (s.f.). *The Talos Principle.* https://store.steampowered.com/app/257510/The_Talos_Principle/

[3] Steam (s.f.). *Inside.* https://store.steampowered.com/app/304430/INSIDE/

[4] Google Play (s.f.). *Candy Crush Saga.* https://play.google.com/store/apps/details?id=com.king.candycrushsaga

[5] Google Play (s.f.). *Monument Valley.* https://play.google.com/store/apps/details?id=com.ustwo.monumentvalley

[6] Steam (s.f.). *The Room.* https://store.steampowered.com/app/288160/The_Room/

[7] Bandai Namco (s.f.). *Very Little Nightmares.* https://es.bandainamcoent.eu/little-nightmares/very-little-nightmares

[8] Bandai Namco (s.f.). *Somos Bandai Namco Europe.* https://es.bandainamcoent.eu/

[9] Gavin Wright (Febrero 2023). *What is augmented reality (AR) gaming?* TexhTarget. https://www.techtarget.com/whatis/definition/augmented-reality-gaming-AR-gaming#:~:text=Augmented%20reality%20gaming%20(AR%20gaming)%20is%20the%20real%2Dtime,global%20positioning%20system%20(GPS)

[10] Yolanda Valery (7 julio 2016). *Que es Pokemón GO.* BBC Mundo. https://www.bbc.com/mundo/noticias-36736858

[11] Greg Kumparak (5 Enero 2021). *Minecraft Earth will shut down in June.* TechCruch https://techcrunch.com/2021/01/05/minecraft-earth-will-shut-down-in-june/

[12] Isabel Romero Gómez (11 Abril 2023). *AR vs VR: Diferencias entre realidad aumentada y realidad virtual.* Onirix. https://www.onirix.com/es/ar-vs-vr/#:~:text=La%20principal%20diferencia%20entre%20la,usuarios%20pueden%20sumergirse%20y%20explorar.

[13] Unity (s.f.). *Plataforma de desarrollo en tiempo real de Unity.* https://unity.com/es

[14] Microsoft Visual Studio (s.f.). *Visual Studio Code - Code Editing. Redefined.* https://code.visualstudio.com/

[15] CorelDraw (s.f.). *software de diseño gráfico, ilustración y técnico – CorelDRAW.* https://www.coreldraw.com/la/

[16] Adobe (s.f.). *Adobe Illustrator | Software de ilustración digital y vectores.* https://www.adobe.com/es/products/illustrator.html

[17] PTC (s.f.). *Software de realidad aumentada (RA) para empresas Vuforia*. https://www.ptc.com/es/products/vuforia

[18] Google (s.f.). *Google Sites*. https://sites.google.com/new?hl=ES