I.E.S LAS SALINAS

PROYECTO FINAL FIN DE GRADO

CURSO 22-23

Aprende Jugando

CICLO FORMATIVO GRADO SUPERIOR

DESARROLLO DE APLICACIONES MULTIPLATAFORMA

Autor: Iván Bernal Lara

Tutor: Oscar Lillo Díaz

RESUMEN

Aprende jugando es una aplicación móvil para dispositivos móviles, la cual está enfocada a niños de entre 6 a 10 años. La aplicación no solo consta de entretenimiento y diversión, sino que también es educativa y ayuda a desarrollar sus capacidades motoras.

Consiste en varios juegos, uno de ellos es el clásico juego de memoria de encontrar parejas, otro juego se basa en pulsar la pantalla para limpiar unas manchas y obtener puntos, y el último juego que se basa en operaciones matemáticas básicas (Sumas y restas).

Para ayudar a facilitar las interacciones dentro de la aplicación y dar un aspecto más amigable el juego está basado en torno a una mascota. Además, también ayudara a ver el juego mucho más divertido y entretenido.

RESUME

Learn playing is a mobile application for mobile devices, which is focused on children between 6 and 10 years old. The application not only consists of entertainment and fun, but it is also educational and helps to develop their motor skills.

It consists of several games, one of them is the classic memory game of finding pairs, another game is based on pressing the screen to clean some blobs and get points, and the last game is based on basic mathematical operations (addition and subtraction).

To help facilitate interactions within the application and give a friendlier look the game is based on a pet. In addition, it will also help to make the game much more fun and entertaining.

Índice

1. JUSTIFICACIÓN DEL PROYECTO

Las nuevas generaciones dan uso a la tecnología cada vez más temprana, entre ellos los más pequeños, y muchos de los padres no saben actuar ante esto.

Muchos jóvenes hacen un mal uso de la tecnología, y es por eso que realizando aplicaciones enfocada a ese tipo de público puede ayudarles indirectamente a cambiar eso, haciendo un entorno no solo más seguro sino más amigable, donde usando las herramientas adecuadas pueden mejorar su conocimiento, mientras se divierten y no ser un medio dañino para ellos.

El objetivo de esta aplicación móvil es ayudar a que eso se convierta en una realidad, proporcionándoles una aplicación más donde pueden aprender y expandir su conocimiento con entretenidos juegos. Intentaremos que sea lo más adaptable posible, para que todo tipo de público con diferencia de edades pueda usarla sin cambiar su simplicidad. No pretendemos que esté únicamente enfocada en ser una aplicación didáctica, sino que también tendrá focos en el tema psico-motriz y lógica entre otros

2. INTRODUCCIÓN

Somos un pequeño grupo de desarrolladores de juegos los cueles tratamos de ayudar a mejorar el entorno digital para los jóvenes. Esta idea se nos ocurrió tras observar noticias sobre el mal uso de los dispositivos móviles y el adelantado uso de la tecnología entre los más jóvenes, tras realizar estudios al respecto nos propusimos realizar una aplicación enfocada en ayudar a mejorar el ambiente que se encuentran los más pequeños al entrar en el mundo tecnológico.

Para este proyecto contamos con la colaboración de Castilla-La Mancha, quien nos ha contratado, gracias a ello tendremos la oportunidad de incluir la aplicación en su paquete de apps gratuitas de su entorno educativo ***Educamos***, además de otras plataformas y tiendas de aplicaciones para que tenga más repercusión.

2.1 Acerca de la organización

Para realizar este proyecto hemos sido contratados por Castilla-La Mancha, trabajaremos juntos en un departamento que estará enfocado a este tipo de aplicaciones infantiles. Los fondos del proyecto han sido suministrados por Castilla-La Mancha. Actualmente este quipo lo formamos 3 personas que aportaremos en cada aspecto del proyecto. Ya teníamos un lugar de reunión, asique lo usaremos como local de trabajo y por ahora usaremos nuestros equipos personales para trabajar en el proyecto, ya que cumplen con los requisitos necesarios para trabajar con ellos. En el desarrollo del proyecto usaremos ordenadores con sistema Windows.

En lo que a seguridad digital respecta, usaremos el antivirus de Windows, ya que es el más seguro, crearemos correos nuevos y utilizaremos un programa para escanear todos los archivos que usemos, para comprobar que están libres de malware y añadiremos una contraseña muy compleja a nuestro equipo para que nadie pueda acceder a ellos. También tendremos seguridad física, como un SAI (Sistema de alimentación contra apagones), para poder guardar el progreso en caso de que se vaya la luz durante el desarrollo, cámaras de seguridad para la vigilancia del local e implantaremos una puerta que solo se abrirá con un codigo en la entrada del local.

2.2 Estudios y planteamientos

Para plantear el proyecto hemos realizado varios estudios y análisis para encontrar el enfoque de nuestra aplicación. Tras analizar los estudios hemos encontrado que la mayoría de niños usan los dispositivos con malos hábitos que pueden ser contraproducentes para su salud tanto físico como psicológicos y muchos de los padres no son conscientes de ello por lo que no añaden las medidas necesarias para evitarlo.

Buscando datos sobre como fomentar un buen uso de la tecnología en niños, nos hemos encontrado muchas medidas como el equilibrar el tiempo de uso de los dispositivos, o evitando la interrupción del sueño. Uno de los datos con más importancia entre otros es el de darles la opción de jugar juegos acordes con sus edades, siendo una gran manera de apoyar su desarrollo cognitivo, o incluso mejorar sus conocimientos utilizando juegos educativos apropiados.

Para contribuir a que eso sea posible optamos por enfocar nuestro proyecto en ese apartado. En la bibliografía se pueden encontrar enlaces a páginas que corroboran esta información.

3. OBJETIVOS

El objetivo principal es aportar una aplicación que sea capaz de mejorar el entorno digital para los jóvenes. Para hacer eso posible hemos propuesto los siguientes objetivos:

- Impulsar el movimiento de las aplicaciones educativas.
- Fomentar el buen uso de la tecnología aportando una aplicación apta para todas las edades.
- Apoyar el desarrollo o mejora de las capacidades del usuario, así como la lógica y tiempo de reacción entre otros.
- Entretener al usuario de una forma creativa.

4. DESARROLLO

4.1 Tecnologías empleadas

Para realizar el proyecto hemos utilizado varios programas y recursos a lo largo del proceso.

ANDROID STUDIO

Para desarrollar la aplicación hemos decidido usar el IDE oficial de Android. El coste es nulo debido a que Android Studio tiene licencia freeware, entre sus principales ventajas respecto a otros IDE es que nos ofrece muchas herramientas que nos facilitarán mucho el trabajo y agilizará el proceso de desarrollo. Además, en el apartado gráfico es muy flexible, utiliza XML para crear la interfaz.

JAVA

Como lenguaje de programación hemos elegido ***JAVA***. Ya que es un lenguaje bastante versátil y es de fácil uso, además tiene una comunidad muy amplia que contribuye a añadir librerías y funciones casi a diario.

GITHUB

También hemos utilizado ***GITHUB*** para tener más controlado el codigo, esta herramienta nos permite poder volver a versiones anteriores de la aplicación de una forma sencilla e ir guardando el progreso. Además, nos puede permitir trabajar en equipo gracias a las opciones que te permite realizar.

GIMP

El programa ***GIMP*** nos ha brindado un gran número de posibilidades para editar y crear los recursos visuales necesarios. Este programa está bajo la licencia pública de GNU (Programa de manipulación de imágenes digitales) por lo tanto es de uso gratuito y no añade costos adicionales.

DIA (Diagram Editor)

Este programa nos brinda una inmensa cantidad de posibilidades hablando de diagramas, nosotros lo usaremos como pizarra para analizar los casos de uso de la aplicación.

TRELLO

Trello será necesario no solo para organizar las tareas, sino que también lo usaremos como herramienta para aplicar la metodología usada en el proyecto que describiremos en el siguiente apartado.

IMÁGENES USADAS EN LA APLICACIÓN

Algunos de los recursos de imágenes usadas en la aplicación, como las manchas del juego invasión de manchas, o los árboles del juego memoria provienen del servicio gratuito PNGWing. También hay recursos del servicio Depositphotos, como la mascota, para ello hemos invertido en un plan Flexible mensual, ya que los recursos son usables con todos los derechos de por vida tras una Suscripción. Los enlaces a las páginas están en la bibliografía.

EFECTOS DE SONIDO

Para dar mejor ambiente a la aplicación usaremos música y sonidos llamativos, para ello usaremos varios servicios que nos lo proporcionan de manera gratuita, estos se pueden ver más detalladamente en la bibliografía.

4.2 Metodología usada

METODOLOGÍA AGILE

Esta se basa en el conjunto de técnicas para gestionar el desarrollo del proyecto de forma flexible y eficaz mediante ciclos de trabajos cortos, reduciendo así los costes de producción y aumentando la productividad del mismo. Dentro de esta metodología nos hemos centrado en Kanban, que se centra en la gestión del desarrollo de una forma visual, siendo así posible visualizar el flujo de trabajo rápidamente y en la limitación del trabajo, es decir, será necesario acabar cualquier tarea empezada tratando de evitar los problemas que surgen cuando queremos realizar muchos trabajos a la vez, causando así la desorganización.

Las fases dentro de Kanban son 3:

- Por hacer: Indica las tareas que están listas para comenzarlas.
- En proceso: Aquí se mostrarán las tareas que se están realizando en el momento.
- Finalizado: Las tareas ya realizadas se pasarán a esta última fase.

Para aplicar correctamente esta metodología usamos la herramienta Trello, ya indicada anteriormente en el apartado de tecnologías empleadas.

4.3 Análisis de requisitos

SISTEMA OPERATIVO

Para la aplicación que desarrollaremos usaremos Android 8.0 Oreo como versión mínima para instalar la aplicación, haciendo así que la compatibilidad sea posible con el mayor número de dispositivos.

INTERFAZ

El diseño de la UI está orientada a niños y jóvenes, hemos decidido que implementaremos una interfaz simple, fácil de usar y apta para todo tipo de usuarios.

Es importante recalcar que la aplicación está centrada en el color azul, porque el color azul aporta tranquilidad, seguridad y confianza así que es un buen método de hacer sentir mejor al usuario visualmente. Además, hemos querido añadirle toques de color naranja, que inspira creatividad y energía.

SONIDO

Un juego sin sonido o música se hace mucho más monótono, para evitar eso, añadiremos música adecuada para un juego de niños y para que no se haga muy pesado al usuario cambiaremos la música dependiendo del juego en el que entremos, de esta forma no solo conseguiremos una aplicación más animada, sino que cada juego se sentirá único y diferente.

COMPLEJIDAD DE LOS JUEGOS

Los juegos de la aplicación deben tener un nivel de complejidad lo más bajo posible, de forma que cualquier persona pueda llegar a entender el objetivo del juego, mientras se mantiene el nivel de entretenimiento para que no se haga aburrido para quien lo juegue más de una vez y para apoyar eso, se tendrá en cuenta que las partidas deben ser sesiones cortas.

Nos apoyaremos en la base de la mascota para dar contexto detallado y conciso del juego antes de empezar a jugar.

DATOS DE LOS USUARIOS

No se recopilará información adicional y datos privados de los usuarios, puesto que queremos realizar la aplicación apta para todos los públicos, asegurando así el cumplimiento de la ley de protección de datos.

IDIOMA DE LA APLICACIÓN

Para que la aplicación tenga mucho más alcance queremos incluir que la aplicación pueda estar tanto en castellano como en inglés, además esto ayudara a los usuarios a aprender inglés.

4.4 Procedimiento del desarrollo

Ya que tenemos una idea general de lo que tratará la aplicación vamos a separar el proyecto en las diferentes fases por las que pasaremos durante el desarrollo.

4.4.1 Planificación del proyecto

Dividir bien las tareas para saber en qué enfocarnos es muy importante usando la metodología Kanban, para ello se han repartido dependiendo la dificultad y complejidad. Para que se entienda mejor hemos creado un diagrama de Gantt tal y como se ve en la imagen:

ACTIVIDAD	HORAS EMPLEADAS
Pantalla de Inicio	4 HORAS
Pantalla menú principal	6 HORAS
Pantalla de dificultad	9 HORAS
Juego Memory	9 HORAS
Juego de pintura	9 HORAS
Juego de preguntas	9 HORAS
Musica e idiomas	6 HORAS
Mascota	4 HORAS

Imagen del diagrama de Gantt para representar la división y gestión de las mismas

4.4.2 DISEÑO DE LA INTERFAZ

Como lo analizamos anteriormente, el diseño de la interfaz tiene que ser simple, fácil de entender y llamativa para captar la atención del usuario. Las interacciones deben estar distribuidas para que en un solo vistazo pueda saberse que se puede hacer sin que sea una sobrecarga de información para el usuario.

Para mantener el jugador activo en los juegos, estos no tendrán botones que realicen una determinada acción, sino que dejaremos que el jugador de forma táctil realice las acciones pulsando una parte de la pantalla o la opción que desean escoger. Para entender más las acciones que puede realizar el usuario en la aplicación se puede ver de una forma clara en el siguiente diagrama de casos de uso:

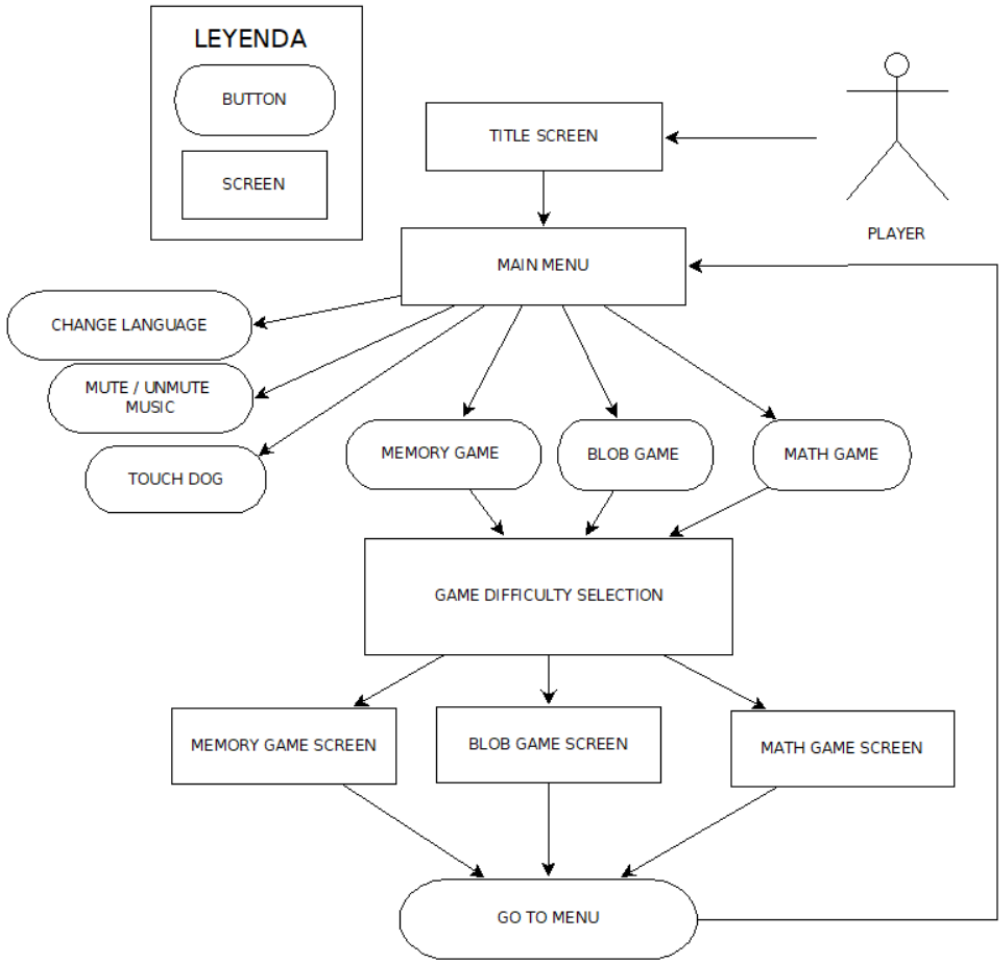


Imagen del diagrama de Casos de uso

4.4.3 DESARROLLO DE LA INTERFAZ

Comenzando el desarrollo de la interfaz, el primer paso fue crear la pantalla inicial, añadimos una pantalla inicial para que la primera impresión la aplicación fuera de una forma divertida, y que el usuario se sienta agradable con la transición al entrar a la aplicación.

Imagen de la pantalla de título de la aplicación

Como se puede apreciar, la pantalla de inicio es muy simple, cumple con el objetivo de no sobrecargar al usuario con mucha información. Esta especialmente recalcado el texto de como proseguir, es decir, pulsando la pantalla.

Tras realizar la pantalla de inicio, proseguimos con la pantalla del menú principal, en ella aparecerán varias acciones que se pueden realizar, cambiar el idioma, silenciar la música en el juego, acceder a los juegos y acariciar la mascota.

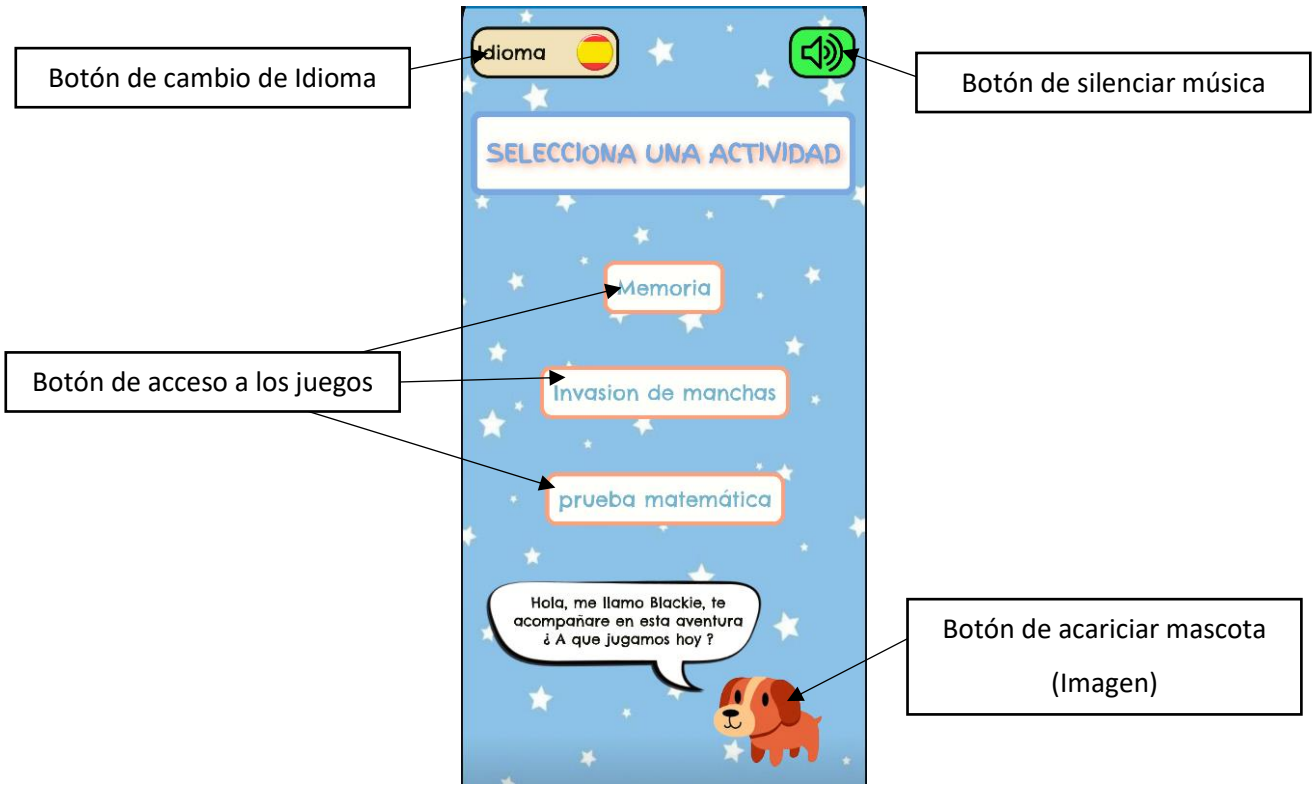


Imagen de la pantalla del menú principal de la aplicación

Las acciones detalladas que puede realizar el usuario en esta pantalla son las siguientes.

- Botón de cambio de idioma: Al pulsar este botón, cambiara el idioma de la aplicación a inglés.
- Botón de silenciar música: Como se indica en la imagen el botón está en color verde indicando que el sonido está habilitado, tras pulsarlo se silenciará la música de toda la aplicación (Esto no afecta a los efectos de sonido, que seguirán escuchándose).
- Botón de acariciar mascota: La acción de acariciar la mascota cambiara el dialogo y la posición de la mascota, esta acción esta "escondida" es decir, que no es intuitivo pulsar a la mascota, y está hecho intencionalmente para que cuando suceda la acción llame el interés del usuario.

- Botón de acceso a los juegos: Estos serán los botones que el usuario pulsará para acceder a los juegos, antes de acceder al juego se dirigirá a la pantalla de selección de dificultad.

Una vez el usuario ha seleccionado el juego se mostrará la pantalla de selección de dificultad, de esta forma le damos a elegir al usuario el nivel de dificultad que quiere jugar en su partida. La pantalla está compuesta por los botones de selección y un botón para comenzar el juego, tal y como se muestra en la imagen siguiente.

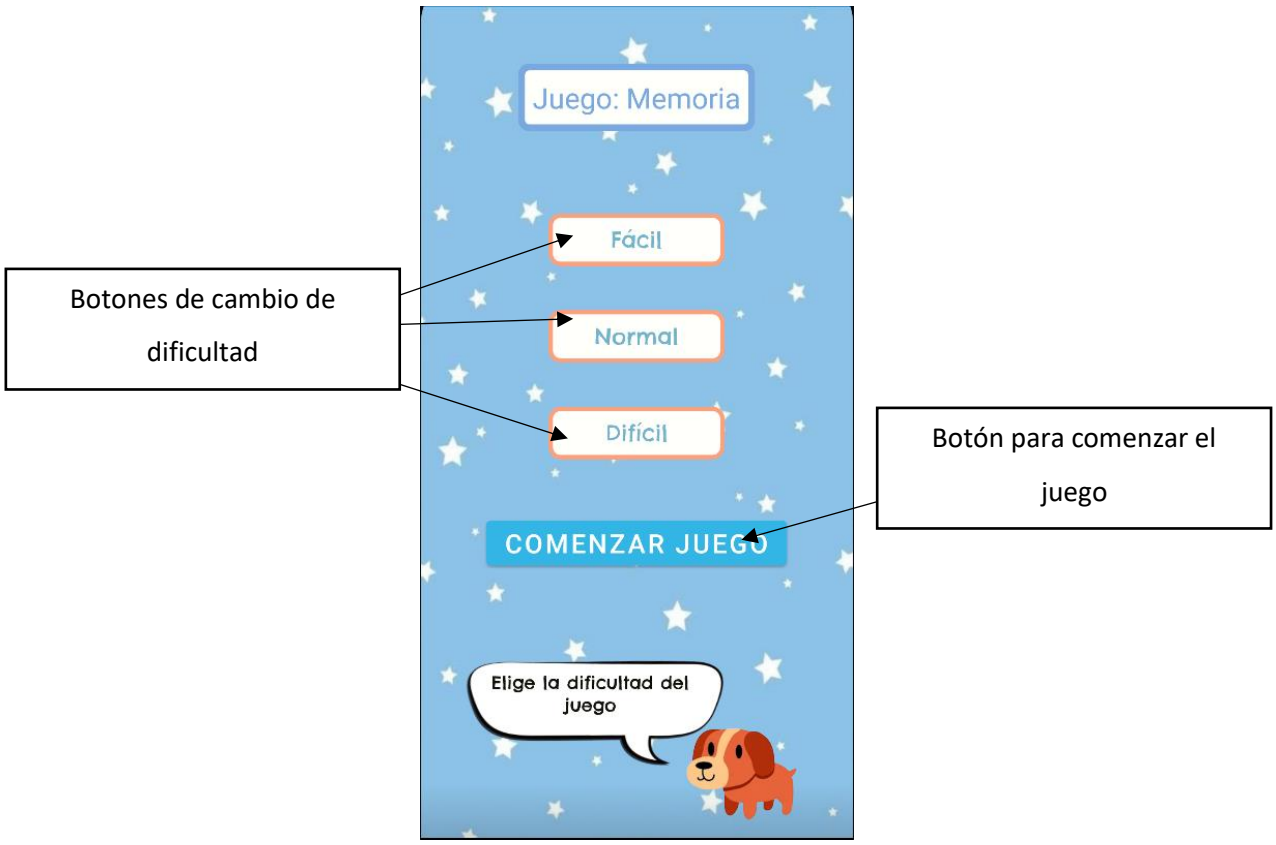


Imagen de la pantalla de selección de juego

Una vez se ha seleccionado el nivel de dificultad se lleva al usuario a la pantalla de juego, todos los juegos tienen interacciones diferentes, pero hemos decidido que tengan un panel de información donde se muestre lo que necesita saber el usuario.

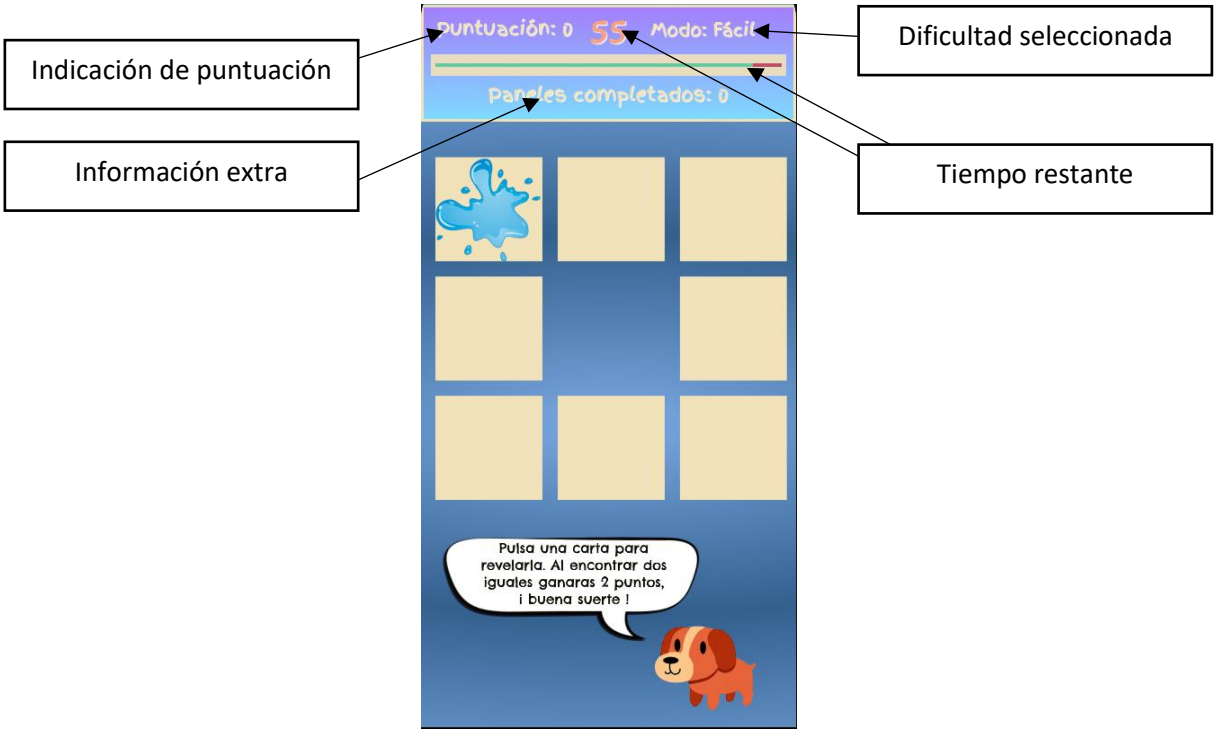


Imagen de la pantalla del juego “Memoria”

En esta pantalla podemos apreciar varios aspectos visuales e interactivos:

- Indicación de puntuación: Este nos indicara la puntuación que llevamos en la partida.
- Dificultad seleccionada: Nos mostrará la dificultad en la que está el juego.
- Tiempo restante: Para indicar el tiempo de juego de una manera visual, hay una barra que se irá reduciendo acorde al tiempo restante, también puedes ver el tiempo restante en texto en el número indicado encima de la barra de tiempo.
- Información extra: La información que se encuentra debajo de la barra del tiempo nos indica información necesaria para el juego, así como los paneles completados. Este cambia dependiendo del juego.

4.4.4 DESARROLLO DEL CÓDIGO

Tras tener una clara idea de cómo se iba a realizar el código y realizar el diagrama de casos de usuario empezamos a desarrollar el codigo del menú principal.

A continuación, se explicarán las partes del código más importantes respecto a cada parte de la aplicación, para entenderlo mucho mejor ver el *Diagrama de clases del anexo 6.2* .

Menú principal

Este apartado hace referencia a la clase **MainMenu**, en el que se encuentran tres funciones importantes:

- Opción de cambio de idioma: Esta opción es un botón, tras pulsarlo la aplicación llamará al método que cambiará al idioma que no esté actualmente como predeterminado. Para ver más a fondo este método puede verse referenciado en el apartado *Accesibilidad - Cambio de idioma* en el Anexo 6.1.
- Ajuste de silenciar la música en la aplicación: La función de silenciar la música en toda la aplicación se basa en una variable booleana estática que funciona como ajuste, cada vez que se pulsa el botón el booleano cambia su estado haciendo así que sea True o False. Si es False, el sonido se desactivará y soltara la música que haya en memoria.
- Botones para seleccionar el juego: Tras pulsar cualquiera de los botones de juego se recogerá el texto del botón para saber cuál juego es e iniciará la actividad de selección de dificultad pasando el texto que acabamos de recoger como argumento, que veremos en el siguiente apartado.

Además de estas dos funciones, se puede acariciar a la mascota, esta función iterará aleatoriamente sobre las acciones predefinidas, luego se cambiará el dialogo y la imagen de la mascota dependiendo de la acción.

Selección de dificultad

Este apartado hace referencia a la clase **DifficultySelection**, la Activity de selección de dificultad recogerá el argumento del que hablamos en el apartado Menú principal (Botones para seleccionar el juego) y lo usara para mostrar al usuario que tipo de juego ha elegido.

El usuario tendrá que elegir una de las tres dificultades mediante sus botones, cada botón hace referencia a una función. Estas funciones se pueden ver más detalladamente en el apartado *Dificultad* del Anexo 6.1.

Tras elegir una de las tres dificultades el usuario debe pulsar el botón comenzar juego, que llamará a la función “start_game” e iterará mediante el texto que se recibió como argumento al iniciar la Activity, dependiendo del texto iniciará un juego u otro como se muestra en la imagen.

```
106     public void start_game(View view) {
107         //When the user start the game we check which activity it will start by the name and
108         // pass the difficulty value
109         MainMenu.menu_music.release();
110         if(mediaPlayer!=null){
111             mediaPlayer.release();
112         }
113         mediaPlayer = MediaPlayer.create( context: this, R.raw.start_game_sound);
114         mediaPlayer.start();
115         if(game_name.equalsIgnoreCase("Memoria")){
116             Intent intent = new Intent( packageContext: this, MemoryGame.class);
117             intent.putExtra(DIFFICULTY_SELECTED, difficulty);
118             startActivity(intent);
119             finish();
120         }
121         if(game_name.equalsIgnoreCase("Invasion de manchas")){
122             Intent intent = new Intent( packageContext: this, BlobGame.class);
123             intent.putExtra(DIFFICULTY_SELECTED, difficulty);
124             startActivity(intent);
125             finish();
126         }
127         if(game_name.equalsIgnoreCase("prueba matemática")){
128             Intent intent = new Intent( packageContext: this, MathGame.class);
129             intent.putExtra(DIFFICULTY_SELECTED, difficulty);
130             startActivity(intent);
131             finish();
132         }
133     }
```

Imagen de la función “start_game” en la clase DifficultySelection

Los juegos

Este apartado esta divido para diferenciar y ver detalladamente cómo funciona cada juego.

Memoria

El juego de memoria se divide en varias clases:

- **MemoryGame**: Es la clase principal de este juego que se encarga de coordinar todas las demás y de hacer de intermediario entre interfaz y código.
- **Card**: Esta clase se encarga de instanciar cada tarjeta visual, para más tarde asignar un valor a cada carta.
- **CardsManagment**: Es responsable de coordinar las animaciones de las cartas, la imagen de cada carta y de comprobar que dos cartas seleccionadas son pareja.
- **MemoryTimer**: Se trata de un Hilo que actuará de reloj del juego, que funcionará como un temporizador empezando desde el número que se le pase.
- **StageValue**: Esta clase se encargará de repartir los valores para todas las cartas, de manera que ningún valor se repite y cada carta tenga una pareja.

MemoryGame recogerá el valor de dificultad al iniciarse y lo asignará al nivel de dificultad, se instancia una clase de **CardManagment**, un nuevo **MemoryTimer** y crea un nuevo tablero con la clase **StageValue**. También llamará a la función que se encarga de iniciar la música (que se puede ver más detalladamente en el apartado *Accesibilidad* del Anexo 6.1). Cuando el usuario pulse la pantalla se ejecuta el método "start_game" que comenzará el **MemoryTimer** (Runable) ejecutándose el método Run que es el temporizador del juego. Cuando el jugador pulse una carta para revelarla, se le pasara el ID de la carta a **CardManagment** que ejecutará una animación a la carta que se le pasa, le dará una imagen respecto al valor que tiene la carta (Para darle un valor visual), esta imagen cambiará cada panel completado (Stage) y guardara el valor, tras pulsar una segunda carta se comprueban si las dos cartas tienen el mismo valor que fue asignado anteriormente en **StageValue** si no tienen el mismo valor (no hacen pareja) se iniciará una animación para esconder las cartas, en cambio, si tienen el mismo valor, las cartas se volverán invisibles y se suman puntos a tu puntuación actual, así hasta acabarse el tiempo de juego y mostrar la puntuación final al jugador, junto a un botón para volver al menú.

Invasión de manchas

El juego invasión de manchas se divide en las siguientes clases:

- **BlobGame**: Es la clase principal de este juego, se encargará de coordinar las demás clases y de comunicarse con la interfaz para dar información al usuario.
- **BlobList**: Es la clase que indicará el identificador de cada mancha y si ya está agregada en el tablero o no.
- **BlobTimer**: Se trata de un Hilo que actuará de reloj del juego, este también se encargará de agregar manchas a la pantalla cada cierto tiempo, e ir decrementando el tiempo que tarda en agregarse hasta llegar al mínimo (1 segundo).

Una vez se recoge el nivel de dificultad elegido se determina cual va a ser el numero total de manchas que se pueden acumular y se asigna un tiempo predefinido que se usara para calcular cuando hay que añadir una nueva mancha al juego. Se inicializa un **BlobTimer**, para que funcione como reloj del juego y vaya añadiendo manchas. Tras ello el juego enlaza cada mancha con un ID y se guardan en **BlobList**. El temporizador añadirá una mancha y disminuirá el tiempo que tardan en reaparecer (para ver esto detalladamente ver el apartado *Invasión de Manchas* del Anexo 6.1), e incrementará el contador de manchas visibles actualmente. El usuario al pulsar una mancha aumentará la transparencia de la misma, de forma que parezca que desaparece, cuando la mancha supere o llegue a la transparencia 0, es decir, que no es visible, la imagen de la mancha se volverá invisible y mediante aleatoriedad se cambiará la posición, después le sumará un punto al jugador. Si se acumula el limite de manchas visibles, o se acaba el tiempo la partida se termina y se mostrará la puntuación al jugador, junto a un botón para volver al menú.

Prueba matemática

El juego prueba matemática se divide en las siguientes clases:

- **MathGame**: Es la clase encargada de gestionar y coordinar las demás clases y comunicarse con la interfaz para dar información al usuario.
- **MathTimer**: Se trata de un Hilo que actuará de reloj del juego, contará hacia atrás empezando por el número que le pases como argumento.
- **Question**: Esta clase se encarga de recoger los datos de la pregunta actual, así como su respuesta.
- **LifeManager**: Esta clase se encargará de almacenar una vida, tiene una variable que indica si la vida en cuestión ha sido perdida o no.

La clase comenzara creando un Array de **LifeManager**, donde se guardarán las 4 vidas y se dará al valor de false a la variable booleana "vida perdida" de **LifeManager** indicando que tiene las 4 vidas todavía. Después de eso se creará una clase **MathTimer** para que haga de temporizador del juego y se llamara al método para que comience la música. Al empezar el juego se llamará al método "next_question" que creará una nueva pregunta y la guardará en una clase **Question** para acceder a ella más tarde y se le muestra la pregunta al usuario (Para saber más detalladamente este proceso véase el apartado *Prueba Matemática* del anexo 6.1). Sabiendo cual es la respuesta correcta, se generarán números aleatorios como respuestas incorrectas y se le mostrará al jugador. El jugador debe pulsar una de las opciones, si esta es correcta ganará puntos y se repetirá la acción de generar una nueva pregunta, si la respuesta es incorrecta se cambiará el estado de "vida perdida" a true en una de las vidas en **LifeManager**, cuando se genera una nueva pregunta se comprueban también las vidas perdidas del jugador, si las 4 vidas tienen el valor de "vida perdida" el juego se acaba y mostrará la puntuación al jugador, junto a un botón para volver al menú. Además, el juego acabara si se acaba el temporizador de **MathTimer**.

Mascota

La mascota tiene un desarrollo a parte de los juegos, está compuesta de unos Sprites (Imágenes que representan personajes u objetos para crear efectos de movimiento) de la mascota y el dialogo de la mascota.

El código de la mascota es un texto plano, el cual al realizar diferentes acciones este cambiará, por ejemplo, en cualquier juego al ganar puntos se elegirá aleatoriamente un nuevo diálogo mediante un “math.random” (Generador de números aleatorios) que estará asociado a una imagen de la mascota acorde con la frase. Este sistema está puesto de manera que puedan tocar más números de los que realmente tienen diálogo, esto está hecho a propósito para que no se haga muy pesado el cambio de dialogo.

En el menú principal también se puede acariciar a la mascota, tiene el mismo código que se usa en los juegos para que aparezca un diálogo aleatorio.

5. CONCLUSIONES

Tras el desarrollo de la aplicación y un profundo análisis, la aplicación cumple con todos los objetivos. Algunos de ellos como fomentar el buen uso de la tecnología e impulsar el movimiento de las aplicaciones educativas podrían haberse mejorado y haber tomado mayor importancia.

5.1 Propuestas de mejora

Tras el análisis, hay varios aspectos mejorables respecto a la aplicación:

- Interfaz: En la interfaz de la aplicación tiene aspectos mejorables, no solo visuales, sino también de diseño, tales como cambiar la selección de dificultad, para que, en vez de ser una pantalla, sea solo un pop-up, y que parezca aún más rápido el acceso al juego.
- Experiencia de usuario: Para mejorar y dar un nuevo objetivo al usuario, se estudiaría la posibilidad de usar una base de datos para proporcionar mas recursos, una puntuación máxima, para saber cuál es el record de cada juego o incluso poder personalizar el aspecto de la aplicación dependiendo del gusto del usuario.

- Realzar los objetivos que no han cumplido todas las expectativas: Se podrían mejorar los detalles de la aplicación para dar un aspecto mas educativo, añadir nuevos juegos relacionados con el aprendizaje o mejoras similares.

6. ANEXOS

6.1 Anexos del código

Accesibilidad

- **Cambio de idioma:** El método “change_language” comprobará el lenguaje actual, que esta guardada en una variable, si el codigo es inglés se cambiará el idioma a castellano, de lo contrario se cambiará el idioma a inglés. Después se soltará la música que hay en memoria y se reiniciará el Activity con el idioma al que queremos cambiar como default.

```
158    public void change_language(View view) {
159        //When the language button is clicked, it will call setLocal with the desired language
160        if(language.equalsIgnoreCase( anotherString: "en")){
161            setLocal("es");
162        }
163        else{
164            setLocal("en");
165        }
166    }
167
168    public void setLocal(String language){
169        //This will change the locale language and restart the activity
170        menu_music.release();
171        LocaleListCompat appLocale = LocaleListCompat.forLanguageTags(language);
172        AppCompatDelegate.setApplicationLocales(appLocale);
173    }
```

Imagen de las funciones que se usan para cambiar el idioma de la aplicación

- **Música**: La función que reproduce la música comprueba primero si la variable booleana en la clase **MainMenu** es True (Se explica la funcionalidad en el apartado *4.4.4 Desarrollo del codigo - Menú Principal*). Después se comprobará si la música ya se está ejecutando, si es así dejará libre el reproductor, tras ello le asignará al reproductor el archivo con la música que queremos iniciar, y para que se repita en bucle le pondremos el parámetro de Looping a true e iniciaremos la música. Para entender mas el proceso se puede ver en la siguiente imagen:

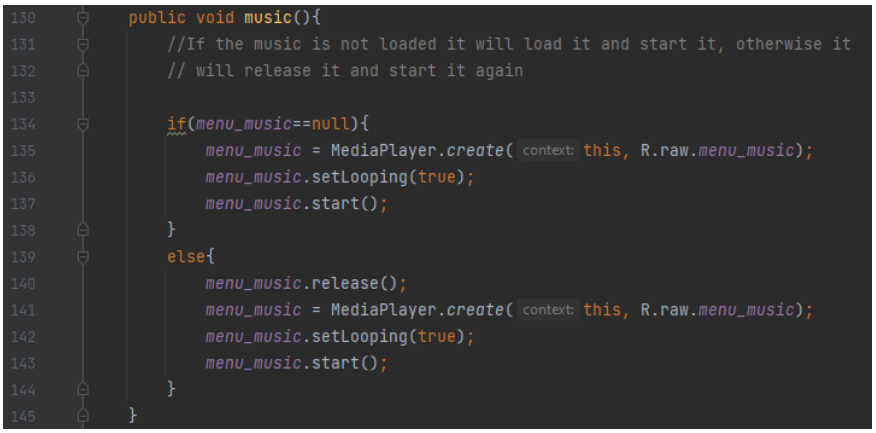


Imagen de la función que se usa para reproducir la música

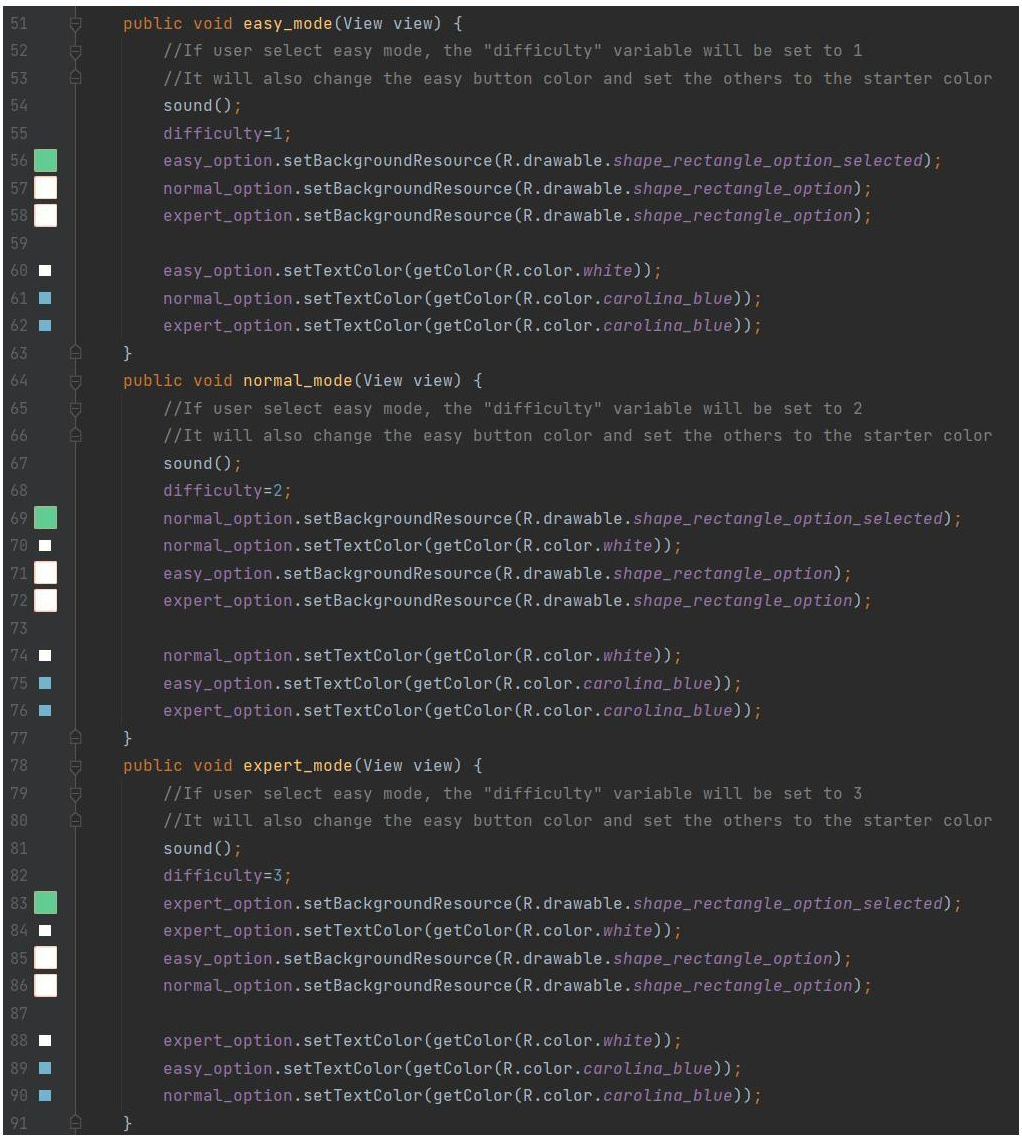
Selección de dificultad

- **Dificultad:** Las funciones de los botones cambiarán la variable en la que se guarda la dificultad, siendo así que si se pulsa el botón cambiará la dificultad tal y como se muestra en la siguiente tabla:

Dificultad	Valor	Función que ejecuta
Fácil	1	“easy_mode”
Normal	2	“normal_mode”
Difícil	3	“expert_mode”

Tabla que representa el valor de cada dificultad en el codigo

Tras pulsar el botón se reproducirá un sonido llamando al método “sound”, tal y como podemos ver en la linea 54 de la siguiente imagen, y también se cambia el aspecto del botón para dar a entender al usuario cual dificultad ha elegido.



```
public void easy_mode(View view) {
    //If user select easy mode, the "difficulty" variable will be set to 1
    //It will also change the easy button color and set the others to the starter color
    sound();
    difficulty=1;
    easy_option.setBackgroundResource(R.drawable.shape_rectangle_option_selected);
    normal_option.setBackgroundResource(R.drawable.shape_rectangle_option);
    expert_option.setBackgroundResource(R.drawable.shape_rectangle_option);

    easy_option.setTextColor(getColor(R.color.white));
    normal_option.setTextColor(getColor(R.color.carolina_blue));
    expert_option.setTextColor(getColor(R.color.carolina_blue));
}
public void normal_mode(View view) {
    //If user select easy mode, the "difficulty" variable will be set to 2
    //It will also change the easy button color and set the others to the starter color
    sound();
    difficulty=2;
    normal_option.setBackgroundResource(R.drawable.shape_rectangle_option_selected);
    normal_option.setTextColor(getColor(R.color.white));
    easy_option.setBackgroundResource(R.drawable.shape_rectangle_option);
    expert_option.setBackgroundResource(R.drawable.shape_rectangle_option);

    normal_option.setTextColor(getColor(R.color.white));
    easy_option.setTextColor(getColor(R.color.carolina_blue));
    expert_option.setTextColor(getColor(R.color.carolina_blue));
}
public void expert_mode(View view) {
    //If user select easy mode, the "difficulty" variable will be set to 3
    //It will also change the easy button color and set the others to the starter color
    sound();
    difficulty=3;
    expert_option.setBackgroundResource(R.drawable.shape_rectangle_option_selected);
    expert_option.setTextColor(getColor(R.color.white));
    easy_option.setBackgroundResource(R.drawable.shape_rectangle_option);
    normal_option.setBackgroundResource(R.drawable.shape_rectangle_option);

    expert_option.setTextColor(getColor(R.color.white));
    easy_option.setTextColor(getColor(R.color.carolina_blue));
    normal_option.setTextColor(getColor(R.color.carolina_blue));
}
```

Imagen de las funciones de los botones para cambiar la dificultad

Invasión de manchas

- **Manchas**: Las manchas (Blobs) son la parte principal del juego. El código enlazará la imagen de una mancha y la agregará a la lista de manchas, que serán guardadas en la clase **BlobList**, y se le asignará un valor extra que nos indicará si es visible o no, al no ser visible significará que este no esta añadido al juego. Para agregar las manchas se usa un complejo sistema que consiste en 3 partes:
    - **Valor necesario para agregar una mancha**: Es el valor que necesitará llegar el contador para que agregue una mancha a la pantalla.
    - **Valor que lleva el contador para agregar una mancha**: Es el valor acumulado que lleva el contador.

- **Valor que se suma al contador**: Es el valor que se suma al contador cada segundo del reloj del juego.

Dependiendo la dificultad del juego el valor que se suma al contador cambia. Si el valor que se suma es mayor llegará antes al valor necesario, por lo que se agregará una mancha con mayor rapidez.

Para que no se haga aburrido en los niveles fáciles el valor que se suma al contador aumentará cada vez que aparezca una nueva mancha, haciendo así que la dificultad sea gradual.

Prueba Matemática

- Generación de nuevas preguntas: La creación de la pregunta es el paso más complejo del juego, es necesario decir que la pregunta será el resultado de una suma o resta, asique necesitaremos cada cosa por separado, comenzaremos asignando 4 variables:
  - Numero 1: Sera el primer sumando o sustraendo de la pregunta.
  - Numero 2: Será el segundo sumando o sustraendo de la pregunta.
  - Tipo: Este indicará el tipo de operación a realizar, suma "+" o resta "-".
  - Respuesta: La variable respuesta contendrá el valor total de la operación.

  Una vez declarados, recogeremos la dificultad, si la dificultad es fácil generará números aleatorios entre 0 y 25 para los dos sumandos y el tipo de operación será siempre una suma, siendo así 50 el número más alto que puede llegar a salir, en cambio, si la dificultad es mayor que fácil (Normal, Difícil) se considerará que la operación pueda ser también una resta, y los números que se generan podrán ser el primero entre 0 y 50 y el segundo entre 0 y 49, de esta forma no llegarán a ser números de 3 cifras, también se ordenarán para que no aparezcan resultados negativos. Para entenderlo un poco mejor se puede ver en la siguiente imagen:

```java
private Question generateQuestion(Integer difficulty){
    //Generates a random question with random numbers and the type ( "+", "-" ) depending
    // on difficulty, it also gets the correct answer and fill the other positions with random
    // numbers
    Integer number1;
    Integer number2;
    Integer answer;
    String type;

    if(difficulty==1){
        number1 = (int) (Math.random()*25);
        number2 = (int) (Math.random()*25);
        type=" + ";
        answer = number1+number2;
    }
    else{
        number1 = (int) (Math.random()*50);
        number2 = (int) (Math.random()*49);
        Integer randomType = (int) (Math.random()*2);
        if(randomType == 1){
            type = " - ";
            do{
                number1 = (int) (Math.random()*50);
            }while(number2 > number1);
            answer = number1 - number2;
        }
        else{
            type = " + ";
            answer = number1+number2;
        }
    }
```

Imagen del método que generará la pregunta

6.2 Diagrama de clases

Para entender mucho mejor la estructura del código podemos ver el esquema de clases a continuación:

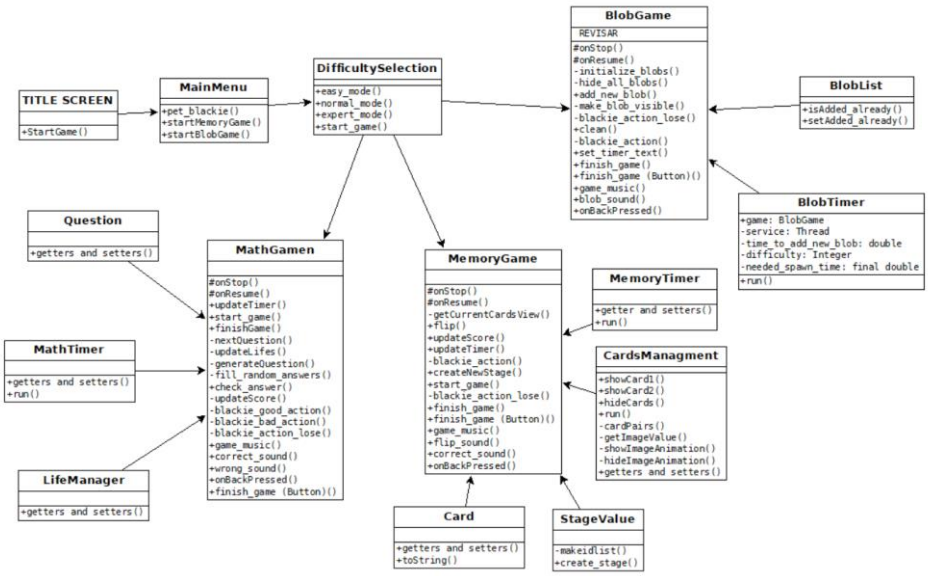


Imagen del diagrama de clases de la aplicación

6.3 Enlace al repositorio de GitHub

Para saber mucho mas a fondo puedes revisar todo sobre este proyecto en el repositorio de GitHub.

También encontrarás la guía de usuario que esta en el archivo README.

- Enlace al repositorio: https://github.com/ivan035/TFG_Ivan_Bernal

7. BIBLIOGRAFÍA

Investigaciones:

https://www.arsys.es/blog/constituir-sociedad-empresa

https://blog.gitnux.com/es/estadisticas-sobre-la-tecnologia-en-la-educacion/

https://dqcertificaciones.eu/los-ninos-y-la-tecnologia-en-la-actualidad/

https://spain.minilandeducational.com/school/la-psicomotricidad-como-motor-del-desarrollo-integral-del-nino/

Música:

https://sfxr.me/

https://mixkit.co/

Imágenes:

https://www.pngwing.com/es

https://sp.depositphotos.com/