APLICACIONES MÓVILES
PRÁCTICA 2
ONE LINE

Alumnos: Carlos Llames Arribas y Gonzalo Guzmán del Río.

Estructura (Assets):

- **JsonDotNet:** Herramienta utilizada para la deserialización de .json. Incluye la funcionalidad para Android, IOS, y windows.
- **Prefabs:**
  - **Button:** Botón que aparecerá en la escena SelectLevel, y que representará un nivel.
  - **Cell:** GameObject que representará una celda del grid. Se compone del bloque gris básico, bloque con color, dirección y pistas.
  - **CoinImage:** Panel que refleja la cantidad de monedas que tenemos.
  - **ExitButton:** Botón de exit en cada escena.
  - **Pointer:** Representación de donde estamos pulsando.
- **Resources:**
  - **/Maps:** Contiene los .json de cada una de las categorías.
  - **/Textures/game:** Contiene carpetas cuyo nombre es cada uno de los colores de las pieles que puede haber en el juego. Basta con añadir una carpeta nueva con los sprites correspondientes y añadir desde el editor, el nombre del color de la piel añadida.

- **Scenes:**
  - **MainMenu:** Escena para seleccionar la categoría a jugar.
  - **SelectLevel:** Escena para elegir el nivel a jugar.
  - **Gameplay:** Escena con el nivel cargado y lista para jugar.

- **Scripts (Clases más detalladas accediendo a cada una de ellas):**
  - **BackButton.cs:** Clase que permite volver a la escena anterior, o quitar el juego.
  - **ButtonLevel.cs:** Clase que tendrá el prefab button utilizado en la escena SelectLevel.
  - **Cell.cs:** Clase que representa el comportamiento de una celda, dándole una dirección, unas coordenadas, activandola o desactivandola...
  - **CoinsText.cs:** Actualiza el texto con las monedas actuales.

    - **2PopulateGrid.cs:** Genera los botones de selección de nivel de la escena SelectLevel.
    - **Managers:**
        - **AdsManager.cs:** Encargada de administrar los anuncios a lo largo del juego.
        - **ButtonManager.cs:** Encargada de administrar los botones del MainMenu.
        - **CellManager.cs:** Encargada de administrar todo el grid de celdas.
        - **GameManager.cs:** Se encarga de inicializar el AdsManager y cargar el progreso del jugador. Administra el funcionamiento del juego y guarda las variables entre escenas.
        - **LeatherManager.cs:** Carga las pieles que utilizarán las celdas y carga una de ellas aleatoriamente.
        - **LevelManager.cs:** Se encarga de gestionar el funcionamiento de un nivel individualmente.
        - **PointerManager.cs:** Representación del puntero del ratón o el táctil de móvil.
        - **UIManager.cs:** Encargado de gestionar el canvas en la escena GamePlay.
        - **ResizeManager.cs:** Encargado del escalado del grid en la escena de GamePlay.
    - **Tools:**
        - **GetCategorySprite.cs:** Devuelve el sprite con el nombre de la categoría actual.
        - **LevelReader.cs:** Lee todos los niveles de una categoría.
        - **Serializer.cs:** Encargada del cargado y guardado de datos.
        - **Timer.cs:** Timer para la cuenta atrás del modo challenge.
        - **TimerToChallenge.cs:** Tiempo restante para poder iniciar un nuevo reto.

- **UnityAds:** Plugin para la generación de anuncios inGame.

Descripción:
El proyecto dispone de 3 escenas individuales. Estas escenas serán el **MainMenu**, **SelectLevel** y **Gameplay**:

MainMenu: Es la primera escena con la que empezará el juego. Dentro del mainmenu, tenemos el GameManager, el cual es un singleton y se encargará de administrar el funcionamiento del juego y del paso de variables entre las diferentes escenas. Se creará solo una vez al iniciar por primera vez esta escena. Además se encarga de cargar los datos guardados, para que el jugador pueda mantener su progreso. Dentro del MainMenu, también tenemos un “grid” de botones, los cuales nos permitirán establecer que categoría es la que queremos jugar, una vez seleccionada, automáticamente se cargará la escena de SelectLevel. Cada botón también reflejará el progreso que llevamos en esa categoría. Pulsando el botón de la casa, arriba a la izquierda saldremos de la aplicación. Los botones de la parte de abajo de la pantalla son meramente estéticos. El encargado de administrar los botones de esta escena será el ButtonManager, dando a cada uno de los objetos un callback específico.

SelectLevel: En esta escena, al igual que en MainMenu, dispondremos de un botón arriba a la izquierda, el cual nos permitirá volver a la escena anterior. En el centro de la pantalla veremos un “grid” de 100 botones. Cada uno de estos botones se corresponde con cada uno de los niveles de la categoría seleccionada desde la escena previa. Estos botones pueden aparecer activados o desactivados si ya nos hemos pasado el nivel o no. Pulsando un botón le diremos al GameManager, que nivel es el que vamos a jugar en la siguiente escena, la de GamePlay.

Gameplay: La escena “más importante” dentro del juego. Recibe la categoría y el nivel que queremos jugar y carga el “grid” de celdas con los datos correspondientes (layout y camino de pistas). La carga de estos datos se realiza a través de la clase LevelReader, la cuál se encarga de leer todos los niveles de la categoría correspondiente, guardados en un .json y deserializarlos. El objetivo es el de completar toda la matriz de celdas, pasando por cada elemento una única vez. Este grid aparecerá siempre centrado en la pantalla. En esta escena, el encargado de administrar el propio gameplay, es el levelmanager, el cuál se comunicará con el gamemanager para decirle que ya ha terminado el nivel. También se comunicará con el leathermanager. Éste es el encargado de la creación de pieles. Cada vez que iniciamos un nivel se seleccionará un color aleatorio. Una piel es el color que se utilizará para el “coloreado” de celdas al pasar por ellas, el color de las pistas y el color del puntero. El puntero es manejado por el PointerManager, desde el propio pointer manager además se gestionará el HandleInput, para así únicamente llamarlo cuando pulsamos sobre la escena.
Una vez terminado el nivel saldrá un panel con el nivel que hemos jugado, con botones para continuar al siguiente nivel o volver al menú principal.
También podemos volver a la pantalla de selección de nivel, pulsando el botón de arriba a la izquierda.
Durante el gameplay, en la parte inferior aparecerán tres botones (borrar el camino, anuncio con recompensa y comprar pista(25 monedas)).
Sin embargo, la escena de gameplay, se puede cargar en modo desafío, con la que el funcionamiento del nivel “cambia”. Al pulsar desde el MainMenu el botón de challenge, el cuál solo puede jugarse pasados 30 minutos desde la última vez, nos lanza directamente a esta escena. La categoría y el nivel seleccionados, serán completamente aleatorios

(ADVANCED-MASTER). Además, en vez de los tres botones de la parte inferior del modo normal, dispondremos de un contador, el cuál representará el tiempo restante para completar el desafío. (Empieza en 30 segundos). Si el tiempo se acaba sin haber completado el nivel, perderemos las 25 monedas que cuesta. Si ganamos recibiremos 50 monedas, las cuáles pueden ser duplicadas viendo un video con recompensa.

Además de todo esto, en todas las escenas se mostrarán anuncios cuando corresponda. Esto se ha hecho a través del plugin de **UnityAds**. Los anuncios mostrados pueden ser con o sin recompensa. Se puede ver un anuncio para conseguir monedas durante el gameplay, para duplicar las monedas conseguidas al finalizar un reto, para iniciar un reto gratuito, o al pasar de niveles.

Finalmente, todos los objetos pertenecientes al juego se escalan según la relación ancho-alto de la pantalla. La resolución referencia es 720x1280 y la mínima aceptada es 1:1. Por decisión de diseño, se ha optado por añadir un scroll a los elementos centrales del canvas, para no perder la vista de esos objetos cuando la resolución tiende a ser cuadrada.

La clase **Serializer** se encarga de guardar el progreso del juego y de cargarlo cuando se inicia. Se guarda el número de monedas, el número de medallas, el progreso del usuario en cuanto a niveles y el timestamp en el que el usuario pulsó el botón de challenge.