**Memoria de la práctica 2**

Jorge Rodríguez García y Gonzalo Sanz Lastra

**Estructura general del proyecto:**

El proyecto está dividido en las siguientes escenas: LoadScene, MenuScene, LevelSelectorScene, GameScene y ChallengeScene, cada una de ellas con un Manager. Los Manager pueden ser accedidos de la forma Manager.instance desde todos los objetos del juego, mientras que los Manager tienen referencia a los objetos que necesiten ser gestionados por ellos, generalmente objetos de UI o el tablero con los tiles.

**LoadScene** es la escena inicial. Carga todos los niveles e instancia el objeto DataManager, que será el encargado de almacenar la información de la aplicación en ejecución y transmitirla al resto de escenas. Además, será el encargado de cargar y guardar los datos de las partidas. De esta forma, al principio de la ejecución de la aplicación cargará los datos guardados, a los que las demás escenas tendrán acceso a través de DataManager. La partida se guardará al quitar la aplicación o cuando ésta salga de foco.

**MenuScene** muestra el menú principal, desde el que se podrá ir a LevelSelectorScene o ChallengeScene.

**LevelSelectorScene** muestra todos los niveles de la dificultad seleccionada y lleva a GameScene con el nivel seleccionado por el jugador, si es que no estuviese bloqueado.

**GameScene** es la escena de juego como tal, con un tablero representando el nivel, etc.

**ChallengeScene** es la escena de reto, similar a GameScene, pero con una cuenta atrás para completarse.

Se ha implementado toda la funcionalidad indicada en el enunciado:

**La visualización de los elementos en pantalla** se adapta con la relación ancho-alto de forma usable y estética hasta resoluciones 1:1. Como característica particular, decir que el diseñador elige desde el editor el tamaño que quiere que tenga el tablero para cada una de las dimensiones (6x5 o 6x8). De esta forma, el marco de respeto que se deja entre el tablero y los límites de la pantalla se elige de forma visual y no desde código: el diseñador decide la escala del tablero en la resolución “ideal” dada en el editor (9:16 a 1080x1920) y sobre esa elección, se hacen los cálculos para que el tablero se adapte a cualquier aspect-ratio y tamaño.

**Leemos los mapas** de un fichero .json, el cual tiene una cabecera indicando el número de dificultades en las que están divididos los niveles, y qué numero de niveles corresponden a cada dificultad. Añadir nuevos niveles no supone cambiar código, basta con ir al .json, escribir un nuevo nivel y decir a qué dificultad corresponde en la cabecera.

**Disponemos de pieles como datos**, elegidas aleatoriamente entre un array de posibles pieles. Para añadir una nueva piel, basta con añadir a un objeto vacío el componente Skin (arrastrando los Sprites que conforman la piel) y hacerlo Prefab. Después, habrá que añadir el nuevo Prefab al array de posibles Skins de BoardManager desde el editor.

**Mantenemos el progreso del usuario**, protegiéndolo contra posibles manipulaciones como aprendimos en clase. Como sal, usamos una cadena de texto (esto podría mejorarse añadiendo a la cadena datos variables, como por ejemplo el número de niveles desbloqueados que tú propusiste).

**Mostramos anuncios al usuario**, como se explica en el enunciado. No hemos hecho que aparezca un anuncio cada vez que se completa un nivel, ya que en el juego original no ocurre y no nos convence por temas de diseño, ya que nos parecen demasiados anuncios que aburrirían al usuario; aunque así lo indica el enunciado. No obstante, añadir esta funcionalidad sería muy sencillo. Aparecen para poder ganar monedas, acceder al reto gratis o duplicar las ganancias del regalo o el reto.

Cuidamos además de que el jugador no pueda interactuar cuando se está proporcionando feedback, comprar pistas cuando están todas visibles, ... en general tener el **juego cerrado**.

Hemos añadido la funcionalidad del regalo, aunque no aparecía nada sobre él en el enunciado. Los botones del menú principal de estadísticas, información, más juegos y quitar anuncios son interactuables pero no han sido implementados.

Los scripts utilizados se pueden ver con más detalle en la documentación adjunta a la entrega y en los propios .cs.