AUTORES

*Daniel Pérez Luque

*Daniel Quintero Bernal

MEMORIA

CLASES MÁS RELEVANTES

Game Manager: Singleton. Gestiona las variables globales del juego tales como los diamantes -moneda freemium-, las estrellas conseguidas por nivel, las estrellas globales y los puntos por nivel.

Además tiene un bool [] de los niveles accesibles para el usuario. El Game Manager tiene una clase dentro, que no es de tipo Monodevelop, que es la clase que serializamos en el método Save del Game Manager

y que luego se carga en el método Load en el Start del Game Manager, si existe.

Por tanto nuestro Game Manager gestiona también el save y el load. La serializacion la hacemos usando BinaryFormatter, generando un stream de bytes, y la guardamos en la PersistentDataPath para que sea una ruta dependiente del dispositivo.

Guardamos partida siempre que obtenemos estrellas nuevas y siempre que desbloqueamos un nivel nuevo automáticamente.

Level Manager: Singleton. Es la clase que gestiona el nivel de juego cargado en ese momento. El Game Manager le da el nº del nivel que tiene que cargar y el Level Manager lo genera usando en LectorTxt. Además, el level manager gestiona los puntos, las estrellas

y las condiciones de ganar o perder la partida. Sirve como gestor de los objetos de juego, ya que tiene unas listas donde se recogen todas las bolas activas, los bloques y los power ups.

Canvas Manager: Singleton. Es la clase que gestiona todo el canvas del nivel. Actualiza los puntos obtenidos, gestiona gráficamente las estrellas logradas, tiene el contador de bolas del spawner y gestiona además

todas las ventanas emergentes tales como la de ganar/perder la partida, la de la compra de power up, compra fallida de Power up, y todos los callbacks de los botones. Tiene un botón de debug inactivo para pasar de nivel rápidamente.

Canvas Menú: Es una clase que gestiona todo el canvas del menú principal. Tiene el logo, los callbacks de los botones entre el que destaca el callback de cargar nivel, único para todos los botones, que carga automáticamente el nivel pulsado siempre que esté desbloqueado.

Visualmente no hay forma de ver que está desbloqueado, pero no te permite entrar si no está desbloqueado.

OrganizaNivel: Clase encargada de colocar los botones de forma particular en "escalera" según las coordenadas locales del canvas.

Además, es la encargada de oscurecer y desactivar los niveles inaccesibles así como de mostrar encima de los niveles desbloqueados el número de estrellas que ha logrado el jugador.

LectorTXT: Es un lector que sabe interpretar cada uno de los 10 niveles que tenemos, aunque podría interpretar más si los txt cumplen con las normas para que la lectura sea correcta.

Cuando quieres generar un bloque, en la primera matriz se establece el tipo del mismo independientemente de si este es bloque o power up. Usamos los mismos IDs que en el juego original. Luego la segunda matriz es la matriz de las vidas de los bloques, pero en nuestra implementacion

para el power up tiene que coincidir el mismo número que su tipo. Es decir, que un bloque de tipo 21 (láser vertical) debería tener "vida" 21 a pesar de no tener vida como tal.

AdsManager: Clase que se encarga de gestionar la aparición de los anuncios mediante el metodo Show() cuando el usuario decide ver un anuncio. Controlará además si el usuario se ha saltado el anuncio antes de tiempo o si lo ha visto en su totalidad, en tal caso se le darán diamantes -moneda virtual-

Scalable camera: Clase que se encarga de cambiar el atributo OrthographicSize de la cámara en función del ratio de ancho y alto deseado. Sirve para ajustar la cámara a distintas resoluciones.

BLOQUES ESPECIALES:

*Hemos implementado el bloque especial de sumar 1, 2 y 3 bolas para el nivel en el que estás actualmente. Pasar de nivel no conserva esas bolas extra.

*Hemos implememententado el bloque especial del láser, tanto horizontal como vertical. Este bloque genera láseres que rompen los bloques de su fila o su columna. Consiste en un Raycast2D en ambas direcciones, a las que apuntan las flechas, desde el centro del objeto.

En estos rayos tomamos la información de todos los bloques golpeados y le decimos al level manager que les reste vida. Este bloque tiene un hijo "rayo" que es el componente visual del bloque. Tiene un sprite inactivo que se pone activo cuando el power up se activa.

Cuando está activo, llama a una corroutina que lo borra tras 0.5 segundos para dar el efecto de rayo fugaz.

POWER UPS:

*Implementamos el power up de generar 4 láseres aleatorios por el mapa. Estos láseres pueden ser horizontales o verticales, y nunca van a instanciarse encima de un bloque.

Cuesta 25 Diamantes y considerando que tiene 100 iniciales lo podrías usar 4 veces antes de quedarte sin diamantes

*El usuario puede ganar Diamantes viendo anuncios en el menú de selección de niveles. No está capado el número de anuncios que puede ver, pero la cantidad que ganas actualmente es 10 con lo que te llevará ver 3 anuncios conseguir 30 diamantes, y así poder permitirte la compra de un powerup.

OTROS:

*El usuario puede hacer doble click (con ratón y con el dedo) para acelerar la partida y que las bolas vayan más rápido. Saldrá una alerta visual simple y breve. Este valor se reinicia por ronda, y por paso de niveles.

*Cuando hayan salido todas las pelotas, saldrá un boton en la barra inferior para devolver las pelotas al spawner. Cuando pasen 3 segundos desde que salga el botón, empezará a parpadear/brillar suavemente para recordar al jugador que no tiene por que esperar a que las pelotas mueran por si solas.