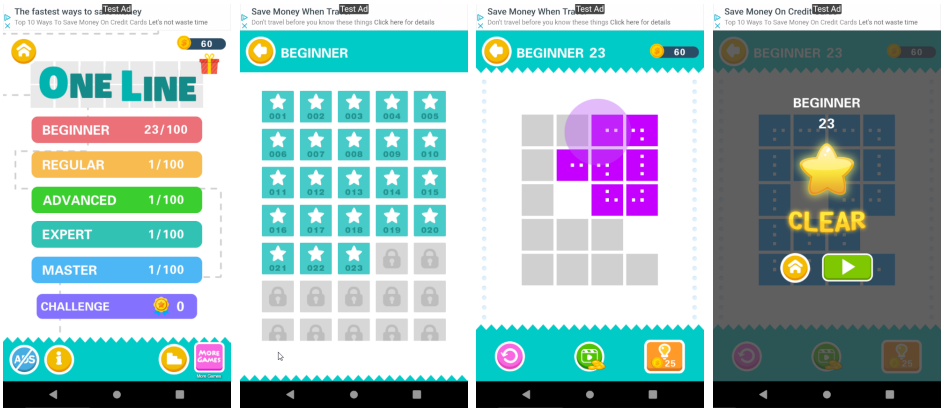
Práctica 2: Clon de *One Line*

Fecha de entrega: 7 de enero de 2020, 23:55

Objetivo: Desarrollo para móvil con Unity. Progreso, publicidad y moneda virtual



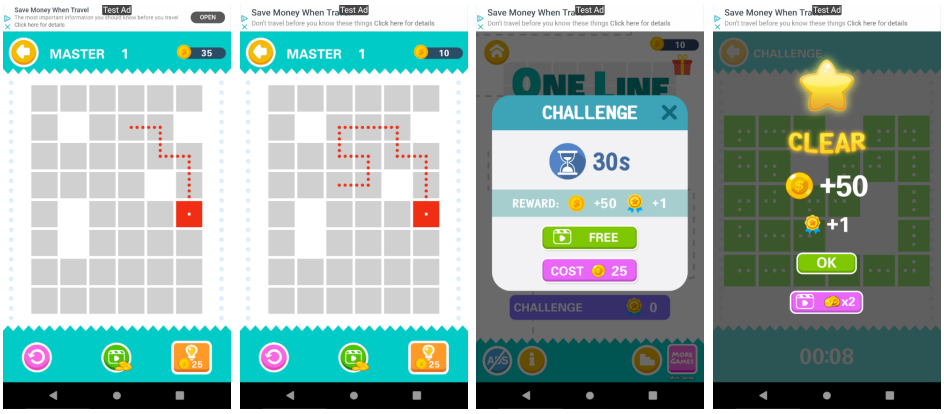
Los puzzles de tipo "*one line*" (también conocidos como *single line* o *one stroke*) consisten en pasar por una serie de puntos sin "levantar el dedo". Hay dos variantes principales. En la primera se debe crear una figura pasando por los puntos potencialmente varias veces creando por tanto un *camino euleriano* sobre el grafo subyacente. En la segunda, se debe pasar por los puntos exactamente una vez, buscando por tanto un *camino hamiltoniano*.

En esta práctica utilizaremos la segunda idea, imitando el juego *One Line*[1] del estudio *2baby*. En él, cada nivel consiste en una cuadrícula de bloques. El jugador debe crear un camino que visite todos los bloques exactamente una vez empezando por un bloque especial, marcado como el punto de inicio, y avanzando siempre por bloques adyacentes (en horizontal o vertical) sin repetir.

Por usabilidad, es posible detener el camino en cualquier momento, pero luego deberá ser siempre continuado en el mismo punto. Si el usuario pulsa sobre un bloque que ya forma parte del camino, se elimina la parte que comienza en él.

[1] `https://play.google.com/store/apps/details?id=com.sencatech.game.oneline`

1. El juego



La práctica consiste en la implementación utilizando Unity de un *clon* del juego *One Line* que incorpore las características habituales en los juegos *freemium* para móvil.

Cada nivel del juego consiste en una cuadrícula de bloques, no necesariamente completa, en el que uno de ellos se ha marcado como el punto de partida. Existen diferentes niveles de dificultad de los mapas en función del tamaño de la cuadrícula. Cada nivel tiene *una única solución*.

El jugador debe crear el camino pulsando sobre los bloques adyacentes o "dibujándolo" sin levantar el dedo. Pese a ser un juego de tipo "*one liner*", es posible detener el dibujo temporalmente, siempre que luego se continue el punto donde se dejó. Si el jugador se equivoca, puede reiniciar el nivel con uno de los botones de la ventana, o pulsar sobre cualquier bloque del camino para que se elimine la sección que comienza en él. En particular, pulsando de nuevo sobre el primer bloque se borrará todo.

El juego marca el camino creado a través de una línea punteada, resaltando además los bloques por los que se ha pasado. Cuando se termina el camino (al levantar el dedo), el juego avanza y se muestra el siguiente nivel.

Los niveles están organizados en categorías de dificultad (*Beginner*, *Regular*, *Advanced*, *Expert* y *Master*), que determinan principalmente el tamaño del mapa. Al principio, el juego muestra un menú con el que el usuario selecciona dicha dificultad, lo que elige implícitamente el lote de niveles a los que jugará. Al seleccionar uno, se muestra la lista de niveles disponibles, y cuáles de ellos han sido ya superados. El juego *mantiene el progreso* entre sesiones, recordando qué niveles han sido ya resueltos, y solo se permite jugar a aquellos que lo han sido, junto con el primero no superado de cada categoría.

Además del progreso, el jugador acumula moneda virtual, que es utilizada para *adquirir pistas*. La compra de una pista superpone sobre el nivel el camino de los 5 primeros bloques (sin contar el de salida) de la solución. Si se compra una segunda pista, se añaden 5 bloques más del camino, y así sucesivamente hasta que se muestra el camino completo. Una pista se mantiene visible mientras se esté dentro del mismo nivel (siempre que no sea ocultada por el camino del jugador). Si se termina el nivel (o se sale de él), las pistas compradas se olvidarán y deberán volverse a adquirir si se quieren ver de nuevo.

La moneda virtual se consigue a través de *vídeos publicitarios recompensados* ("*rewarded videos*") que el usuario puede pedir ver para recibir unas monedas a cambio. También se pueden superar *retos*, que consisten en niveles que deben ser superados en menos de 30 segundos. Una vez superado un reto, no puede jugarse otro hasta que no pasen al menos 30 minutos.

Categoría	Tamaño	Espacio disponible
Beginner	5×5	6×5
Regular	6×5	6×5
Advanced	6×6	6×8
Expert	6×7	6×8
Master	6×8	6×8

Tabla 1: Tamaño de los niveles

2. Detalles del juego

Visualmente, existen varias *pieles* para los niveles, lo que añade un poco de variación visual. Cada vez que se comienza un nivel, la aplicación elige aleatoriamente la piel.

Los niveles se guardan *en ficheros de texto*, con una primera parte indicando el "mapa" con los bloques, y una segunda parte con el camino de la solución. Por comodidad y versatilidad, todos los niveles de cada categoría irán, concatenados, en su propio fichero de texto.

El tamaño de los niveles (número de bloques de ancho y alto) se va incrementando en función del nivel de dificultad. En el juego de *2baby*, los niveles de cada categoría tienen todos el mismo tamaño, salvo en el caso del *Beginner*, que tiene unos niveles iniciales más pequeños que el resto.

El tamaño en pantalla de los bloques se adapta para que entre el mapa completo, aunque, de nuevo en el juego de *2baby*, solo hay dos configuraciones, eligiéndose una u otra dependiendo del tamaño inicial (tabla 1). En cualquier caso, el nivel aparece siempre centrado en el espacio disponible.

Comprar una pista cuesta 25 "monedas virtuales". Desde la ventana de juego es posible solicitar ver un vídeo recompensado que proporciona 20 monedas.

Jugar a un reto tiene un coste de 25 monedas, aunque se puede evitar el pago viendo un vídeo. Cuando se supera un reto en el tiempo previsto, se concede el logro (que también se mantiene persistente) junto con 50 monedas. Es posible duplicar el número de monedas viendo otro vídeo.

3. Requisitos de la implementación

El juego se implementará en Unity, y tendrá que funcionar en móviles y tablets en orientación vertical (retrato). Al menos deberá:

- Incluir toda la funcionalidad descrita. En caso de duda, puedes utilizar el juego original como referencia.
- Adaptar correctamente la visualización en pantalla en función de la relación de aspecto (ancho-alto) del dispositivo.
- Leer los mapas de ficheros de texto.
- Disponer de varias pieles *como datos*, de modo que añadir nuevas pieles *no* suponga cambiar código.

- Gestionar los niveles (y sus lotes) también *como datos*, de modo que añadir nuevos mapas o niveles de dificultad *no* suponga cambiar código.

- Mantener el progreso del usuario (niveles desbloqueados y retos superados) y la cantidad de moneda virtual acumulada de tal forma que sea difícil romperlo y modificarlo externamente accediendo a los ficheros en el dispositivo.

- Mostrar anuncios al usuario (con UnityAds) para *monetizar* el juego. Se mostrará un anuncio al superar cada nivel además de los vídeos recompensados lanzados por el usuario.

En general el juego deberá estar *cerrado*, es decir proporcionar una experiencia de juego fluída y pulida. Se deberá tener cuidado, por ejemplo, con no permitir al usuario interactuar con el juego mientras se está proporcionando *feedback*, o comprar una pista si ya es visible todo el camino. Desde el punto de vista de programación, el proyecto deberá estar ordenado, el código documentado y bien estructurado.

4. Consejos de implementación

Empezad programando la funcionalidad básica con la lógica de un nivel, sin preocuparos de la carga desde mapas, el progreso o la cantidad de moneda virtual disponible para poder comprar o no una pista. En la primera versión, incluso, podéis ignorar el ajuste el mapa a la pantalla.

Aunque tengáis que refactorizar código con frecuencia, tened siempre *un proyecto jugable* al que ir incorporando funcionalidades adicionales poco a poco. Probadlo con frecuencia en diferentes resoluciones de pantalla y relaciones de aspecto.

Pensad si necesitáis algún *manager*, para qué, y cuál va a ser el modelo general de comunicación y coordinación entre los objetos del juego. Decidid qué escenas vais a tener y cómo os vais a comunicar entre ellas.

5. Partes opcionales

Siempre que los requisitos básicos de la práctica funcionen correctamente y estén bien implementados, se valorará positivamente la incorporación de características adicionales:

- Facturación integrada de bienes digitales: que el usuario pueda eliminar los anuncios completamente tras un pago en la plataforma, o adquirir moneda virtual para pistas.

- Conexión con redes sociales: permitir al usuario compartir su avance en Twitter o Facebook a cambio de monedas.

- Creación de un *editor de niveles* (fuera del juego) que ayude a los diseñadores a saber si el nivel que están creando tiene una única solución, y cuál es.

6. Entrega de la práctica

La práctica debe entregarse utilizando el mecanismo de entregas del campus virtual, no más tarde de la fecha y hora indicada en la cabecera de la práctica.

Sólo un miembro del grupo deberá realizar la entrega, que consistirá en un archivo `.zip` con el proyecto completo de Unity eliminando los ficheros temporales. Se añadirá también un fichero `alumnos.txt` con el nombre completo de los alumnos.

Dentro del `.zip` incluid un documento con una breve descripción de la estructura general del proyecto, tanto desde el punto de vista de los ficheros como de las clases, componentes y su relación entre ellas. Indicad las características del juego que habéis implementado y cómo lo habéis hecho, incluyendo las posibles ampliaciones opcionales.

El `.zip` deberá tener como nombre los nombres de los integrantes del grupo con la forma `Apellidos1_Nombre1-Apellidos2_Nombre2.zip`. Por ejemplo para el grupo formado por Miguel de Cervantes Saavedra y Santiago Ramón y Cajal, el fichero se llamará `DeCervantesSaavedra_Miguel-RamonYCajal_Santiago.zip`.