Práctica 2

Videojuego en un ecosistema Android

Objetivo: Mejorar el juego de nonogramas para móviles Android, Práctica 1, aplicando mecánicas y estrategias que permitan monetizar el juego.

1. Monetización:

Para maximizar tus ingresos, es necesario considerar la opción de implementar una estrategia de monetización flexible que se adapte a tu público y mercado objetivo. Los distintos públicos pueden tener preferencias diferentes de aplicaciones pagadas, compras directas desde la aplicación, suscripciones, anuncios y otros tipos de comercio electrónico.

En el modelo freemium damos al jugador acceso a nuestra aplicación de manera gratuita por lo que para monetizar el videojuegos es necesario implementar otras estrategias que permitan obtener ingresos. Dentro de la variedad de juegos que podemos encontrar en la Play Store podemos observar que una gran mayoría de los juegos que siguen este modelo obtienen sus ingresos de anuncios y compras internas.

Esta práctica se centra en implementar mecánicas típicas en los juegos de móviles para poder llegar a monetizar y hacer rentable nuestro juego de nonogramas.

2. Descripción del juego a implementar:

Modos de juego

La pantalla de título ahora debe permitirnos acceder a dos modos de juego. “Juego rápido” y “Modo Historia”.

- **Juego rápido:** Esta opción nos permite jugar a un nivel generado de manera aleatoria (lo que tenemos de la práctica 1).
- **Modo Historia:** Consiste en un conjunto de categorías y niveles en los que únicamente podremos avanzar al siguiente si hemos completado el nivel actual. Los niveles y categorías en este modo son fijos. Para poder tener este modo tenemos que guardar cuantos niveles ha desbloqueado el jugador hasta el momento y crear una interfaz que nos muestre las categorías y niveles disponibles, diferenciando entre bloqueados (no pueden jugarse) y desbloqueados (podemos volver a acceder a ellos en cualquier momento).

En estos niveles del modo historia, deben representarse imágenes pixeladas (una abeja, una casa, un árbol, un faro, etc.). En cuanto a las categorías tenemos 2 opciones y hay que implementar al menos una de ellas:

- Categorías por dificultad: cada categoría tendrá X niveles de una misma dificultad / tamaño, y según avance el jugador completando los niveles de la categoría actual se desbloquearan nuevas categorías con mayor dificultad / tamaño de los tableros. Ejemplo:
  - Categoría fácil, 20 con tableros de 5x5. Al completar los 20 niveles se desbloquea la categoría intermedia con 20 niveles de 10x10.
- Categorías por temática: cada categoría representa una temática y los niveles de cada categoría aumentaran de dificultad. Ejemplo:
  - Categoría Bosque (niveles que representan entidades que encontramos en el bosque) con 20 niveles donde los 5 primeros son tableros de 5x5, los 5 siguientes de 10x10, otros 5 de 15x15 y por último 5 niveles de 20x20. Completar los 5 primeros niveles de una categoría te da acceso a la categoría siguiente, que podría ser por ejemplo “Mar” (niveles que representan entidades que encontramos en el mar).

Anuncios

Existen diferentes anuncios que podemos mostrar en nuestra aplicación, en este caso querremos usar dos tipos:

- **Banners**, anuncios básicos que aparecen al inicio o al final de la pantalla
- **Rewarded**, anuncios que permiten reproducir videos que son recompensados.

La pantalla de título debe mostrar ahora un anuncio de tipo banner en la parte inferior o superior de nuestra ventana (Google Ads).

Los vídeos recompensados serán opcionales para el jugador, dándole ventajas sobre el juego si deciden verlos. Para más información sobre cómo recompensamos al jugador leer la subsección siguiente, *Retención y recompensas*.

Retención y recompensas

Queremos que nuestros jugadores jueguen el mayor número de días posibles, por ello es importante implementar métodos que incentiven al jugador a seguir jugando. Los perfiles de jugadores en móviles son muy variados y vamos a centrarnos en 2:

- Jugadores a los que les gustan los puzzles que proponen retos. Para estos jugadores implementaremos un **sistema de vidas** con el que castigarlo si falla en sus decisiones. Si el jugador marca una casilla que es errónea perderá vida, obligando a comenzar de nuevo si pierde todas. Ahora usaremos "**long touch**" para

marcar las casillas que no forman parte de la solución y el “touch normal” únicamente servirá para marcar las celdas como solución.

- Jugadores a los que les gusta expresarse y no necesariamente les gusta que les compliquen la vida. Para estos jugadores crearemos un conjunto de características que les permite customizar la vista del juego y la paleta de colores del tablero.

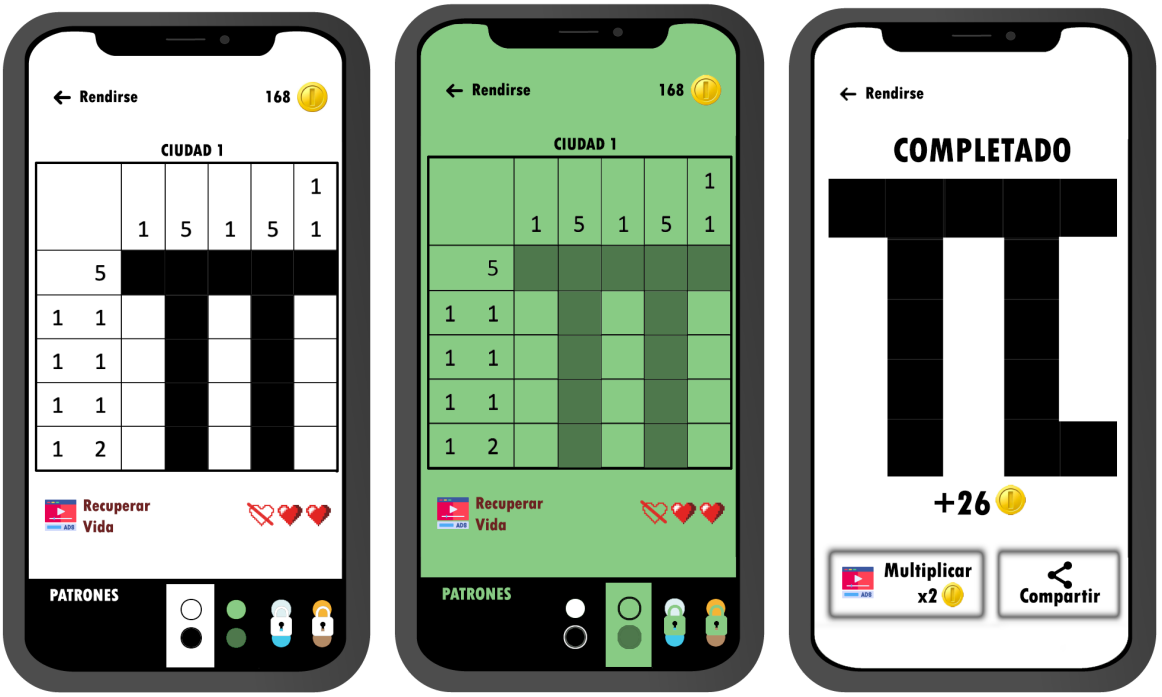
No queremos perder a jugadores por tener dificultades para completar un nivel por lo que es importante permitir a este tipo de jugadores recuperar vida durante los niveles. Y a su vez queremos que los jugadores interesados en expresarse no tengan todas las opciones de customización desde un principio, si no que tengan que ganarlo según avanzan en el juego. Por ello crearemos un sistema de recompensas que permite conseguir las paletas de colores y a su vez que nos de la opción a ganar también vidas.

Las **recompensas customizables** (paleta de colores, apariencia) podrán conseguirse de las siguientes formas (obligatorio implementar al menos una de las dos que se proponen):

- Con un sistema de moneda del juego de forma que puedan desbloquearse por una cantidad específica de esa moneda. La moneda se ganará al ver anuncios recompensados y/o por completar niveles.
- Desbloqueandose al completar categorías o niveles específicos.

Por otra parte las vidas podrán recuperarse en medio de un nivel mediante (obligatorio implementar al menos una de las dos que se proponen):

- La visualización de anuncios recompensados
- Si se implementa un sistema de monedas, gastando las monedas para comprar vidas en medio de un nivel.



Interacción social

Nos interesa que nuestros jugadores hagan publicidad del nuestro juego para llegar a más usuarios, así que nunca está de más dar la opción de compartir al jugador cuando consigue algún logro como pasarse un nivel complicado. Para estos casos, nuestra vista al completar un nivel tendrá un botón que permite a los jugadores **publicar un tweet** sobre sus avances **o** enviar un **mensaje por telegram/whatsapp** a sus amigos.

Notificaciones push

Android nos permite notificar a los usuarios con mensajes, aunque ser muy pesado puede ser contraproducente. Debido a la gran cantidad de juegos diferentes a los que los usuarios tienen acceso pueden olvidarse de la nuestra, para evitar esto, intentaremos **recordar al jugador que nuestra aplicación existe** si sigue instalada y además lleva tiempo sin entrar. Para ello lanzaremos una notificación push. Además avisaremos al jugador de que si entra ahora le recompensamos de alguna forma: cosmético que no tenga o una cantidad de monedas en particular pueden ser una opción.

Sensores

Hay videojuegos que utilizan los sensores para obtener información del entorno. Los sensores más comunes utilizados en dispositivos móviles son aquellos que permiten obtener la posición geolocalizada del jugador y el giroscopio junto al acelerómetro para saber si el móvil está produciendo un giro. El primero es utilizado en juegos geoposicionados mientras que el segundo es utilizado por algunos juegos de conducción para simular un volante.

En esta práctica debéis **recoger la información de al menos un sensor** del dispositivo móvil para transformarla en información ante la que el juego actúa de una manera determinada. La mecánica a implementar depende de vosotros, y puede ser algo tan simple como modificar el menú principal del juego en función de la luz ambiente o del continente donde se está jugando.

3. Requisitos de la implementación

Para el desarrollo se hará uso de Android Studio, utilizando un único proyecto con uno o varios módulos, pero esta vez solo tendremos los propios de Android. El desarrollo será nativo para **Android** y dejaremos de lado la parte de escritorio.

La aplicación se debe adaptar a cualquier resolución de pantalla. Y permitiremos jugar tanto en horizontal como en vertical. En el caso del juego **en horizontal adaptaremos el layout** para que el tablero sea el centro de nuestra pantalla (esta vez no dejaremos bandas a los lados) y los botones se reposicionarán a los lados para dejar que el tablero ocupe el máximo tamaño posible.

Si el jugador cambia de aplicación o cierra nuestro juego en medio de un nivel, debemos permitir que el jugador **recupere el estado de su tablero** al volver a iniciar el juego. No queremos que pierda progreso.

Hay que tener en cuenta que la interfaz de usuario, cómo se disposicionen los botones y las diferentes interfaces es cosa vuestra siempre y cuando se cumplan los requisitos de funcionalidad pedidos. Y las imágenes mostradas son una idea de cómo puede presentarse la información, y además no es la mejor forma.

Por último, acordaros que no queremos que el jugador pueda modificar su progreso si tiene rooteado el móvil, así que debemos añadir al menos una capa de **seguridad al progreso y al conjunto de monedas** que lleva ganadas el jugador **siempre y cuando estas se guarden en archivos que el jugador pueda leer y modificar fácilmente** si tiene acceso a ello. Para ello es necesario usar un **hash o encriptar** los archivos donde se guarde esta información para evitar/detectar modificaciones.

4. Instrucciones de entrega

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

Sólo un miembro del grupo deberá realizar la entrega, que consistirá en un archivo .zip con el proyecto completo de Android Studio eliminando los ficheros temporales. Se añadirá también un fichero **alumnos.txt** con el nombre completo de los alumnos y un pequeño **.pdf indicando la arquitectura de clases y módulos de la práctica, así como un pequeño documento** que explique las vistas implementadas, que recompensas puede conseguir el jugador y cómo, y para qué se ha usado al menos un sensor del móvil.

Para aprobar la práctica es obligatorio que la representación se **adapte al tamaño de la ventana/pantalla**. También es necesario que no haya errores de compilación para aprobar.

5. Evaluación de la práctica

Es obligatorio que la aplicación se ejecute en Android adaptándose a diferentes resoluciones aprovechando al máximo el tamaño. A la hora de corregir la práctica se tomará como referencia la configuración y software de los laboratorios. Los pesos en la evaluación de la práctica serán los siguientes:

- **Funcionalidad y código - 60%**. En este apartado entran por ejemplo la arquitectura planteada, las funciones creadas, limpieza de código, nombres usados, código comentado, estructuras de datos utilizadas, consumo de recursos…
- **Mecánicas, dinámica de juego - 25%**. En este apartado se valorarán las mecánicas implementadas, las recompensas que se dan al jugador por ver anuncios y entrar diariamente así como la interacción con aplicaciones externas.
- **Apartado gráfico y experiencia de usuario - 15%**. Fluidez del juego, coherencia de la interfaz, de los textos y del sonido…

Podrá pedirse a los miembros de un grupo que defiendan la práctica frente a los profesores de la asignatura de manera individual para comprobar el trabajo realizado. Si durante la defensa no se demuestran conocimientos sobre el trabajo presentado el miembro

correspondiente del grupo puede obtener menos nota que sus compañeros, llegando a suspender..

Copiar código de la práctica conlleva el suspenso directo de la asignatura.

Bibliografía

- Beginning Android Games, Third Edition, Mario Zechner and J. F. DiMarzio, Apress, 2016.
- Developing games in Java, David Brackeen, Bret Barker, Lawrence Vanhelsuwe, New Riders.
- Documentación Android
- Monetización Doc Google