UNIVERSIDAD DE EXTREMADURA

Escuela Politécnica

Grado en Ingeniería Informática en Ingeniería del Software

Trabajo Fin de Grado

BreakSquare: Juego para smartphone Android

UNIVERSIDAD DE EXTREMADURA

Escuela Politécnica

Grado en Ingeniería Informática en Ingeniería del Software

Trabajo Fin de Grado

BreakSquare: Juego para smartphone Android

Autor: José Miguel Rodríguez Hernández

Tutor: Juan Manuel Murillo Rodríguez

PRÓLOGO

El objetivo que tiene esta documentación es explicar la elección, definición e implementación de la aplicación Break Square que es un juego para dispositivos (Smartphone y tabletas) Android.

La elección de este proyecto para el Trabajo Fin de Grado surge a raíz de querer dedicarme profesionalmente a la creación de video juegos para dispositivos móviles. En la primera reunión con mi tutor Juanma, le presenté el juego que estaba desarrollando por mi cuenta y le comenté si habría la posibilidad de mejorarlo y presentarlo como TFG, Juanma me dijo que sí y aparte, me animó a que crease un modo de juego multijugador basado (si era posible) en comunicaciones BLE (Bluetooth Low Energy).

El objetivo de este proyecto es crear un juego completo que tenga los mismos detalles que cualquier juego desarrollado por una empresa pueda tener, centrándonos en el uso de una tecnología de comunicación que no fue pensada para su implementación en juegos.

La realización de este proyecto me ha parecido una gran experiencia ya que he puesto en práctica conocimientos adquiridos en el Grado durante los 4 años y además he adquirido nuevos conocimientos en la parte que me ha tocado investigar. Creo que la tecnología utilizada me será muy útil en un futuro sea o no en el campo de los videojuegos.

Por último, desde aquí, quisiera dar las gracias a todas esas personas (familia, amigos y profesores) que me han enseñado, apoyado y ayudado en todo lo que he necesitado desde que empecé en el Ciclo Formativo de Grado Medio “Explotación de Sistemas Informáticos” pasando por el Ciclo Formativo de Grado Superior “Desarrollo de Aplicaciones Informáticas” y terminando (por ahora) en el Grado en Ingeniería Informática en Ingeniería del Software. En especial, quiero dar las gracias a mi madre (Loli), a mi padre (Joaquín) y a mis hermanas (Rocío y Fátima) por haberme soportado y apoyado en todo incondicionalmente para llegar hasta aquí, cosa que nunca imaginamos pero que así es. ¡MUCHAS GRACIAS!

ÍNDICE GENERAL DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

Este documento, recoge la información de todo el proceso realizado para llevar a cabo mi Trabajo Fin de Grado. Dicho trabajo consiste en crear una aplicación móvil (juego) para dispositivos móviles, y el documento recoge desde cómo surgió la idea, hasta como se trabajó en ella, desde que se empezó a diseñar y a modelar, hasta su implementación, y puesta en marcha, pasando por cada una de las etapas necesarias en todo proyecto software.

Este proyecto trata de tocar todos los detalles que puede tener un juego desarrollado por una empresa cualquiera como pueden ser los pagos in-app, publicidad, diferentes modos de juego (multijugador incluido), leaderborads, etc.

Los objetivos de este proyecto son varios, desde crear un juego completo para un sistema operativo muy extendido como Android, hasta investigar y comparar las diferentes posibilidades de comunicación inalámbrica, acabando haciendo uso de una tecnología que en principio no fue creada para implementarla en juegos, pero que se demostrará, que puede ser utilizada para videojuegos con unas necesidades concretas, y el ahorro de batería, será mucho mayor. La tecnología nombrada anteriormente y utilizada será una comunicación BLE (Bluetooth Low Energy) que será gestionada con la utilización de Beacons.

Como resultado de todo el proyecto, se ha creado la aplicación Break Square, un juego individual o multijugador que se centra en crear una experiencia de juego simple y adictiva minimizando el uso de batería. El juego consiste en ir eliminando una serie de cuadrados que tenemos en la parte inferior de la pantalla a través de otros cuadrados (más pequeños) que irán cayendo de la parte superior de la pantalla. La mecánica del juego consiste en pulsar la pantalla para ir parando los cuadrados que van bajando con el objetivo de parar estos últimos, dentro de los cuadrados que están en la parte inferior de la pantalla, de esta forma se irá incrementando la puntuación y se alcanzará el objetivo, que dependerá del modo de juego al que se esté jugando (clásico,

por niveles o en equipo) que será eliminar el mayor número de cuadrados o ir superando cada nivel propuesto.

El nivel de consecución de los objetivos marcados ha sido alto, ya que se han cumplido todos los objetivos marcados, aunque en momentos determinados dadas las circunstancias y la tecnología empleada, se hayan tenido que modificar en mayor o menor medida algunos de dichos objetivos, ya que la tecnología utilizada requiere que sea aplicada en juegos con unas características determinadas y que se detallarán en su parte correspondiente del documento.

A modo de conclusión y como experiencia realizando el proyecto, se ha conseguido que la aplicación citada sea funcional y estoy muy satisfecho con el trabajo realizado. La aplicación es mejorable y se pueden pulir varios detalles, pero me ha servido para poder aplicar los conocimientos adquiridos en el grado y obtenidos durante la investigación realizada. Dicha investigación me ha llevado a conocer mejor la funcionalidad e implantación de la tecnología utilizada y que está emergiendo y siendo muy utilizada en grandes proyectos. Estoy seguro de que los conocimientos adquiridos en este proyecto, me serán de mucha utilidad en un futuro.

1. INTRODUCCIÓN

En este capítulo inicial del proyecto se tratará de dar una visión general sobre cómo está el mercado de los juegos en los dispositivos móviles y cuales han sido las motivaciones para crear este proyecto.

En primer lugar se hablará sobre la evolución de la tecnología y los juegos con respecto al mundo de los dispositivos móviles sufrida en estos últimos años, a continuación también se hablará sobre el internet de las cosas (IoT), seguidamente daremos algunos detalles de cómo la industria del videojuego está apostando cada vez más por la creación de videojuegos para dispositivos móviles, también daremos unas pinceladas de cómo las empresas obtienen beneficios a través de dichos juegos, y por último, se hablará sobre el sistema operativo Android, plataforma en la que correrá nuestro proyecto. En segundo lugar, expondré en qué consiste este proyecto y cuales han sido las motivaciones para llevarlo a cabo. Para finalizar, se mostrará el juego realizado y su funcionamiento.

1.1 EVOLUCIÓN DE LA TECNOLOGÍA

En la actualidad, la mayoría de la población del planeta utiliza Smartphone o Tableta (según un informe de eMarketer, en 2017 se logrará alcanzar un 69,4% de personas que utilizan Smartphone). Hace 10 años, los móviles se utilizaban para llamar, enviar mensajes y jugar de vez en cuando (¿Quién no ha jugado al Snake alguna vez?) pero con la evolución de los móviles a los Smartphone y su fusión con internet, la utilidad que le damos a dichos dispositivos, ha cambiado. Se sigue llamando (al fin y al cabo, es para lo que empezaron a hacerse) pero ahora casi no se envían mensajes de texto (me refiero a SMS, WhatsApp es muy utilizado), eso sí, podemos hacer infinitas cosas más, como, por ejemplo: consultar nuestro correo, ver archivos PDF, editar textos, hacer fotografías, jugar a juegos online, visitar una página web, realizar pagos, y muchas cosas más.

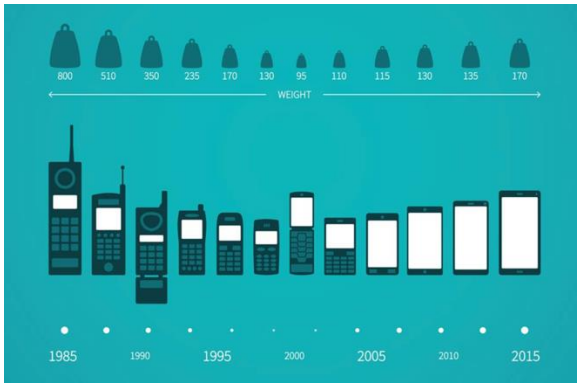

ILUSTRACIÓN 1: EVOLUCIÓN DE LOS SMARTPHONES

FUENTE: HTTP://GEEKFENCE.COM/WP-CONTENT/UPLOADS/2016/01/MOBILE-PHONE-EVOLUTION.JPG

La tecnología avanza a pasos agigantados y centrándonos en nuestro tema, los juegos, podemos decir exactamente lo mismo, han avanzado a una velocidad aplastante en muy pocos años, esto es debido a que las características de los dispositivos han ido mejorando y a que cada vez más gente utiliza su Smartphone para jugar, tanto es así que actualmente, se utilizan más que las videoconsolas o los ordenadores.

Con la aparición de los Smartphone también han aparecido algunos problemas, uno de ellos es la duración de la batería, y aunque cada vez más, se están realizando mejoras con respecto a este problema, tanto creando baterías de mayor durabilidad y aumentando la eficiencia de las aplicaciones,

nada tiene que ver la duración de una batería de ahora con una batería de antaño (de las que duraban 8 o 10 días perfectamente). Por eso, es muy importante elegir bien una tecnología u otra dependiendo de qué necesitemos en nuestro juego e intentar que este sea lo más eficiente posible.

En nuestro caso, como hemos visto en páginas anteriores vamos a realizar una comunicación inalámbrica, esta comunicación ser hará para el modo de juego multijugador y se basará en una comunicación bluetooth a través de beacons.

1.1.1 INTERNET DE LAS COSAS.

Cada vez es más fácil acceder a internet desde cualquier lugar del mundo, con ello, aumentan los dispositivos que se pueden conectar a la red, intercambiar información entre dichos dispositivos y generar una información específica en un momento determinado. Empezaron a conectarse los ordenadores, se continuó por los teléfonos móviles y actualmente se puede llegar a conectar desde una televisión hasta un microondas o nevera entre otros. A esto es a lo que llamamos internet de las cosas (IoT por sus siglas en inglés “Internet of Things”), a la red formada por cualquier aparato que comparte algún tipo de información. Para conectar dichos dispositivos, solamente hará falta integrar un chip de pocos milímetros en cualquier objeto que se nos pase por la cabeza para poder transmitir información a través de él constantemente. Por ejemplo, en nuestro juego, se hace uso de BLE (Bluetooth Low Energy) para transmitir información, crear una red e interactuar con otros jugadores de la misma partida. También se hace uso de internet (wifi o a través de datos) por una parte, para enviar las puntuaciones y compararlas con otros usuarios (que conocemos o no) que han jugado a nuestro juego, y por otra parte, para mostrar publicidad, que será uno de los modelos de negocio en nuestro juego.

ILUSTRACIÓN 2: INTERNET DE LAS COSAS

FUENTE: HTTPS://I.BLOGS.ES/44688D/P6-3/2560_3000.PNG

1.2 LA INDUSTRIA DEL VIDEOJUEGO.

Con la llegada de los móviles y sobretodo los Smartphone, nace una nueva forma de jugar a los videojuegos y nuevos tipos de jugadores. A día de hoy, hablar de la industria del videojuego no es hablar de horas y horas delante del ordenador o la videoconsola, sino que es hablar de gente de todas las edades y todo tipo de condición social jugando al Candy Crash mientras van en autobús, esperan en una cola, o tienen tiempo libre. Antes, para jugar necesitábamos hacerlo en un ordenador, máquina recreativa o videoconsola, ahora sólo basta con tener un móvil a mano y ¿quién no tienen un móvil?

La utilización de los juegos en los móviles le está ganando el terreno a las videoconsolas y los ordenadores, tanto es así, que actualmente se generan más ingresos a través de juegos para dispositivos móviles que a través de juegos para ordenadores y videoconsolas. Según un estudio reciente de la consultora Superdata, en la industria de los videojuegos se ha generado 91.000 millones de ingresos durante el pasado año 2016, de los cuales, 41.000 millones se han obtenido a través de los videojuegos móviles.

ILUSTRACIÓN 3: INGRESOS GENERADOS POR LA INDUSTRIA DEL VIDEOJUEGO.

FUENTE: HTTPS://WWW.SUPERDATARESEARCH.COM/MARKET-DATA/MARKET-BRIEF-YEAR-IN-REVIEW/

1.2.1 MONETIZACIÓN DE UN APP MÓVIL.

Actualmente, existen varias formas de obtener ingresos en una aplicación móvil, en este apartado vamos a dar una visión general sobre algunas de ellas.

- Aplicación gratuita con anuncios: posiblemente la forma más sencilla de obtener ingresos, aunque no es la forma de conseguir un mayor número de ingresos. Consiste en mostrar publicidad (banner, pantalla completa, etc.) en un momento/lugar que no sea molesto para el usuario.
- Aplicación freemium: La aplicación es gratuita pero dentro de ella habrá funcionalidades o elementos (en el caso de los juegos) que se podrán comprar. Este es uno de los modelos más exitosos para obtener buenos ingresos.
- Aplicación de pago: Lo de toda la vida, pagas por la aplicación y disfrutas de ella.
- Aplicación con una combinación de las anteriores: Hay multitud de aplicaciones que combinan varios modelos de los anteriores, por ejemplo, hay aplicaciones que te muestran publicidad y esa publicidad la puedes eliminar si pagas por ello.

1.3 EL SISTEMA OPERATIVO ANDROID.

Android es una plataforma de código abierto (open source) que está basada en Linux. En un principio se creó para dispositivos móviles, pero a medida que

ha ido creciendo su utilización, se ha ido expandiendo a otros dispositivos como televisiones o relojes. Tanto se ha expandido que tiene la cuota de mercado más alta con respecto a sus competidores.

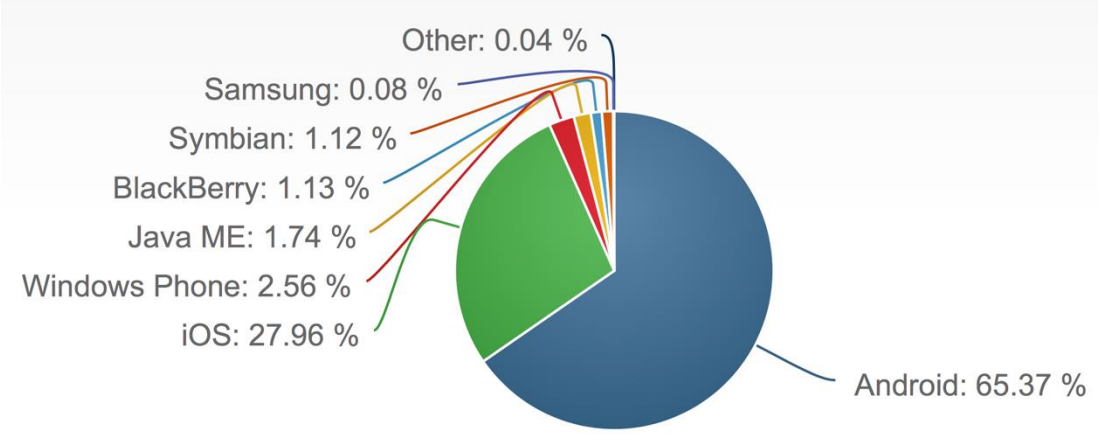

ILUSTRACIÓN 4: CUOTA DE MERCADO DE ANDROID Y SUS COMPETIDORES EN 2016

FUENTE: HTTPS://WWW.NETMARKETSHARE.COM/OPERATING-SYSTEM-MARKET-SHARE.ASPX?QPRID=8&QPCUSTOMD=1&QPSP=2016&QPNP=1&QPTIMEFRAME=Y

Uno de los motivos que ha llevado a Android hasta donde está, es que Google ha apostado por ella desde el principio y su posterior compra después, incitando así a que la gran mayoría de fabricantes de dispositivos móviles hayan optado por añadir a sus dispositivos dicho sistema operativo.

Android dispone de varias versiones, la última es la versión 7.1 (Nougat) pero esto no quiere decir que sea la versión más presente en los dispositivos, al revés, de las últimas 7 versiones, es la menos utilizada. Esto se debe a que todos los dispositivos móviles no han actualizado a la nueva versión bien por ser muy reciente o bien porque no presentan la potencia suficiente para soportarla.

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	1.0%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.1%
4.1.x	Jelly Bean	16	4.0%
4.2.x		17	5.9%
4.3		18	1.7%
4.4	KitKat	19	22.6%
5.0	Lollipop	21	10.1%
5.1		22	23.3%
6.0	Marshmallow	23	29.6%
7.0	Nougat	24	0.5%
7.1		25	0.2%

Data collected during a 7-day period ending on January 9, 2017.
Any versions with less than 0.1% distribution are not shown.

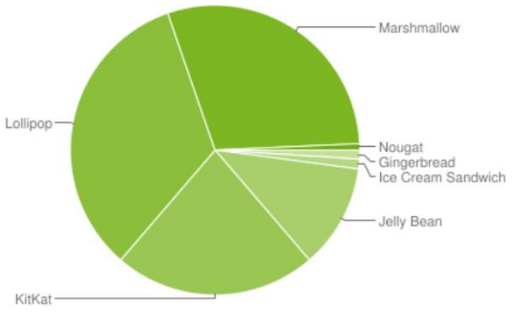

ILUSTRACIÓN 5: VERSIONES DE ANDROID DESTACADAS Y SU PRESENCIA EN LOS DISPOSITIVOS

FUENTE: *HTTPS://DEVELOPER.ANDROID.COM/ABOUT/DASHBOARDS/INDEX.HTML*

1.4 MOTIVACIONES Y OBJETIVOS.

En este nuevo apartado, expresaré las diferentes motivaciones y objetivos que me han llevado a hacer este proyecto.

1.4.1 MOTIVACIONES.

Las motivaciones que me han llevado a hacer este proyecto, son las siguientes:

1. En un futuro me quiero dedicar profesionalmente al desarrollo de videojuegos para dispositivos móviles ya que lo considero un nicho de mercado bastante amplio y en el que más me divierto.
2. En la carrera no hemos visto ningún temario relacionado con la creación de videojuegos así que con este proyecto me he visto obligado a investigar por mi propia cuenta sobre ello.
3. Android es la plataforma de móvil más utilizada y sobre la que hemos trabajado en la carrera.

4. Me resultaba interesante investigar hasta qué punto una tecnología no creada para implementar en videojuegos, podría ser útil para ello.
5. Nunca he trabajado sobre comunicaciones bluetooth y lo veía bastante interesante.
6. No existe ninguna temática de juego igual a la que he creado, algo difícil hoy en día a la hora de crear un juego.
7. Creo que crear este proyecto, me vendrá muy bien para encontrar trabajo en un futuro puesto que toco la mayoría de las cosas (en mayor o menor medida) que tiene hoy en día cualquier juego de cualquier empresa.
8. Aprender el funcionamiento de una tecnología que se está haciendo cada vez más popular y sobre la que están trabajando grandes empresas.

A continuación, paso a describir los objetivos propuestos a la hora de crear la aplicación.

1.4.2 OBJETIVOS.

En este apartado describo una serie de objetivos que se pretenden alcanzar con la creación de este Proyecto Fin de Grado.

Por una parte, está el objetivo principal del proyecto que es desarrollar un juego funcional para dispositivos móviles o tabletas que:

- Se asimile (en cuanto a funcionalidad y tecnología utilizada) a cualquier juego desarrollado por una empresa, con esto me refiero a:
  - Tenga un diseño intuitivo.
  - Añada publicidad.
  - Añada compras en la aplicación.
  - Haya varios modos de juego.
  - Integre leaderboard.
  - Conecte a usuarios a través de una red social.
- Utilice una tecnología inalámbrica reciente.
- Sea jugable de forma offline.

- Rediseñar el juego que ya tengo desarrollado para que el código sea lo más limpio y reutilizable posible.

De forma paralela también hay otra serie de objetivos que surgen de los anteriores:

- Investigar a cerca de las opciones de comunicación inalámbrica que hay disponibles en el entorno de Android.
  - Buscar tecnologías que puedan ser candidatas para aplicar al proyecto.
  - Comparar dichas tecnologías.
  - Indicar en qué aspectos sería mejor utilizar una tecnología u otra.
- Aplicar al proyecto los conocimientos adquiridos durante la carrera.
- Crear una aplicación que minimice el uso de la batería lo máximo posible.

1.5 APLICACIÓN

ILUSTRACIÓN 6: LOGO DE BREAK SQUARE

Break Square es un juego sencillo y adictivo para dispositivos (móviles y tabletas) Android, ideal para pasar un buen rato. La temática de Break Square 2 es sencilla. Hay dos tipos de cuadrados, los rojos y pequeños, y los grandes. Los cuadrados rojos y pequeños estarán colocados de forma horizontal en la parte superior de la pantalla y los cuadros grandes formarán una matriz cuadrada de X cuadrados que estarán situados en la parte inferior de la pantalla. Los cuadrados rojos irán cayendo uno a uno a velocidad y orden aleatorio y si pulsas sobre cualquier parte de la pantalla se pararán. El juego consiste en pulsar la pantalla y parar los cuadrados rojos pequeños, dentro de los cuadrados grandes azules, para eliminar estos últimos.

Para entender mejor la mecánica del juego, puedes ver el video de presentación de una version anterior: https://youtu.be/6N4Q7k-AyHM

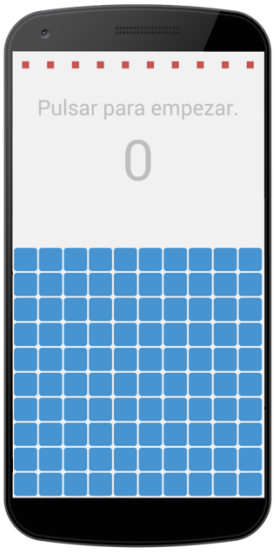

ILUSTRACIÓN 7: VISTA DE LA PANTALLA PRINCIPAL EN EL MODO DE JUEGO CLÁSICO.

Break Square 2 tiene 3 modos de juego:

1. Modo de juego clásico: Consiste en ir eliminando tantos cuadrados como puedas para obtener la máxima puntuación hasta que el jugador falle.
2. Modo de juego por niveles: Consiste en ir superando una serie de niveles que tiene el juego, y que irán aumentando la dificultad a medida que se vaya avanzando. Actualmente consta de 36 niveles.
3. Modo de juego en equipo: Es como el modo de juego clásico, pero con la diferencia de que en la misma partida los cuadrados se irán eliminando entre varias personas y varios móviles diferentes hasta que un miembro del equipo falle.

1.6 CONCLUSIÓN

Como conclusión, he de decir que creo que este primer punto de introducción ha resultado satisfactorio. En él, he dado una vista general de lo que es el mundo del videojuego en los móviles, la forma que tienen las empresas de obtener ingresos y algunos datos relevantes sobre los juegos y Android. También he definido cuales son mis motivaciones, que no han sido pocas y mis objetivos, que tampoco han sido pocos. Por último, he mostrado una parte del juego y su funcionamiento.

Ahora bien, ¿qué vamos a ver en el siguiente punto?: En el siguiente punto ayudaré a entender mejor en qué consiste el proyecto. Para entenderlo mejor, realizaré un análisis de mercado mostrando que aplicaciones hay similares y para cada una de ellas, mostrar cómo funcionan y ver las diferencias y similitudes que hay con la que yo propongo.

2 ESTADO DEL ARTE

Una de las cosas más importantes a la hora de hacer popular un juego es su publicidad, no lo digo yo, lo dicen los expertos. Pero, de nada sirve tener una buena publicidad si el juego en sí, no es bueno o no está bien estructurado, por eso se ha realizado este estudio de mercado, para ver los puntos en los que mi juego destacará con respecto a los de la competencia, al fin y al cabo, se trata de destacar, bien creando algo nuevo o mejorando lo existente.

En este capítulo 2 del proyecto se va a mostrar bajo un análisis de varios juegos existentes en la Play Store, en primer lugar, veremos cómo funcionan, sus cosas buenas y malas, en qué se parece al Break Square y en qué se diferencia. Seguidamente dedicaremos un sub-apartado a hablar de la temática del juego y si existe alguno con la misma temática. Para continuar, hablaré sobre la tecnología utilizada en el juego y si se ha encontrado algún juego que aplique la misma tecnología. Por último, se hablará sobre el público objetivo de dicho juego.

Los análisis y comparaciones anteriores nos ayudarán a entender mejor en qué consiste nuestro proyecto y cuáles serán sus puntos fuertes a la hora de

compararse con algunos de sus competidores una vez lanzado a Google Play. Las comparaciones no se harán de la temática del juego en cuestión, si no de los detalles de los que dispone el juego en general como pueden ser los anuncios, leaderboard, compras integradas, etc. También he de decir que se hará hincapié en que los juegos analizados tenga un modo de juego multijugador, ya que es uno de los puntos fuertes de nuestro proyecto y en el que se utiliza la tecnología bluetooth investigada y seleccionada.

La categoría a la que pertenecen los juegos analizados, es la misma categoría a la del juego creado, dicha categorías son los juegos casuales, pero, ¿qué es un juego casual? un juego casual en general es un juego en el se parte de una lógica de juego muy sencilla que se va repitiendo en los diferentes niveles a medida que aumenta la dificultad. En ellos no se necesita una historia y el tiempo que se le dedica jugándolo suele ser menor, normalmente son pasajeros.

2.1 ANALIZANDO Y COMPARANDO

Como reza el título, en este apartado se realizará un análisis y comparación de juegos que pertenecen a la misma categoría que al que se ha creado en este TFG, para ver los puntos fuertes y débiles de cada uno, así nos haremos una idea de qué tiene bueno/malo el Break Square y poder solucionarlo para que sea mucho más competitivo. El primer juego que se va a analizar, se llama “TIC TAC TOE” y el segundo juego, se llama “Dominó: Juega gratis”.

2.1.1 TIC TAC TOE

Tic tac toe es un juego para dispositivos Android con más de 100.000 descargas en Google Play (números a lo que no se llegan tan fácilmente). Su temática es sencilla, se trata del juego del 3 en raya de toda la vida. En él existen 3 modos de juegos; 1 jugador, 2 jugadores y el juego por bluetooth.

Link de descarga: https://play.google.com/store/apps/details?id=com.ilumnis.xofree

ILUSTRACIÓN 8: VISTA PRINCIPAL JUEGO TIC TAC TOE

Tic tac toe, no presenta una interfaz muy llamativa, pero puede ser válida y la navegación por el juego en sí es bastante sencilla, por lo que se podría decir que es usable.

2.1.1.1 DIFERENCIAS DE TIC TAC TOE Y BREAK SQUARE

- No dispone de publicidad ni de compras integradas. Desconozco si este juego pertenece a una empresa o a un usuario que lo ha realizado por hobby, pero sea lo que sea, no está obteniendo beneficios de esta aplicación, lo cual, considero un error ya que con el número de descargas que tiene, con algún modelo de negocio, podrían obtener ingresos.
- No dispone de ranking ni estadísticas sobre los usuarios, algo que podría hacer que la competencia entre dichos usuarios aumentase y aumentase su número de descargas.
- Es un juego conocido desde antes de que existiesen los teléfonos y ordenadores. Lo cual hace que sea un juego propenso a ser descargado y jugado, ya que es muy entretenido.
- Tiene un apartado que me ha gustado bastante; es una sección en los que se muestran otros juegos realizados por la misma empresa o

persona para animar a que también sean descargados. Esta sección podría ser una ampliación de nuestro juego.

ILUSTRACIÓN 9: TIC TAC TOE VISTA DE UNA VINCULACIÓN BLUETOOTH

2.1.1.2 SIMILITUDES DE TIC TAC TOE Y BREAK SQUARE

- Es un juego muy simple y la navegación por él, es muy sencilla.
- Es un juego multijugador basado por turnos.
- Ambos utilizan una comunicación móvil basada en el bluetooth para el modo de juego multijugador con la diferencia de que Tic Tac Toe usa bluetooth clásico, lo cual hace que haya que emparejar los dispositivos, a lo que hay que añadir que consumo que hace de la batería es mucho mayor. Con Break Square no pasaría eso, ya que utiliza BLE y Beacons.

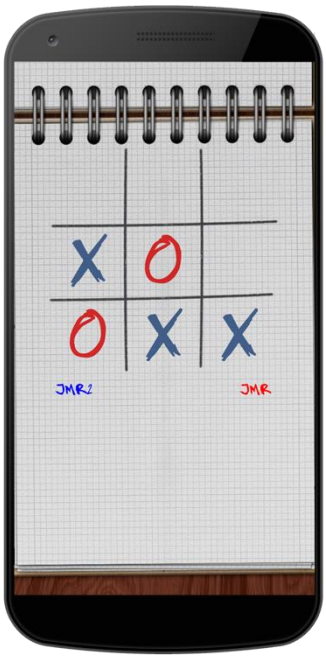

ILUSTRACIÓN 10: TIC TAC TOE VISTA DE UNA PARTIDA

2.1.2 DOMINÓ

Dominó es un juego para dispositivos Android con bastantes más descargas que el anterior, son más de 5.000.000 de descargas en Google Play. Al investigarlo, me he dado cuenta de porqué supera en tantas descargas al anterior. Su temática también es sencilla y es un juego de antaño conocidos por todos. En el juego existen varios modos de juegos: Jugar contra la máquina en diferentes modalidades de juego, jugar contra amigos y jugar contra desconocidos.

Link de descarga: https://play.google.com/store/apps/details?id=air.com.jogatina.domino.android

ILUSTRACIÓN 11: JUEGO DOMINÓ: VISTA PRINCIPAL

2.1.2.1 DIFERENCIAS DE DOMINÓ Y BREAK SQUARE

- Como se ha comentado anteriormente, consta de modo de juego multijugador, pero para ello es necesario tener una cuenta de Facebook para jugar con tus amigos, lo que supone un problema ya que, si uno de los dos usuarios no tiene Facebook, la partida no se podrá llevar a cabo. Sí se puede jugar contra desconocidos en el modo de juego multijugador sin necesidad de Facebook.
- Para el modo de juego multijugador al depender de Facebook o de la plataforma del juego, se necesita conexión a internet, así que en el momento que no se disponga de dicha conexión, la partida multijugador no se podrá llevar a cabo.
- Tiene estadísticas de los jugadores en la partida, pero éstas estadísticas no pueden ser visualizadas por otros jugadores, lo que hace que pierda emoción al igual que la aplicación anterior al no haber competencia más allá de la partida.
- Tiene demasiada publicidad, consta de anuncios de banner y a pantalla completa, aunque hay que decir que no es molesta. Break Square sólo tendrá publicidad a pantalla completa cada dos o 3 partidas.

ILUSTRACIÓN 12: JUEGO DOMINÓ: VISTA DE LA PETICIÓN PARA CONECTAR CON FACEBOOK.

2.1.2.2 SIMILITUDES DE DOMINÓ Y BREAK SQUARE

- El juego dispone de publicidad, lo que será una fuente de ingresos para la empresa desarrolladora.
- Tiene modo de juego multijugador en equipo y por turnos.
- La navegación por el juego es sencilla e intuitiva, lo que hace que la experiencia de usuario sea bastante buena.
- Tiene un diseño muy trabajado y agradable. En este sentido el diseño de Break Square es peor ya que yo no he sido formado en diseño de aplicaciones y tampoco es un tema muy influyente en el proyecto.

ILUSTRACIÓN 13: JUEGO DOMINÓ: VISTA DE UNA PARTIDA.

2.1.3 CONCLUSIÓN DE LOS JUEGOS ESTUDIADOS CON RESPECTO A BREAK SQUARE.

A modo de conclusión de los juegos estudiados, he de decir que podrían mejorar en algunos aspectos, aunque el aspecto más importante que debería mejorar y por el que se caracteriza Break Square es aplicar la tecnología de comunicación bluetooth BLE.

El primer juego (Tic Tac Toe), tiene comunicación bluetooth para el modo multijugador, pero esta comunicación no es BLE, lo que hace que el consumo de batería (a parte de lo que consume el juego en sí) sea mucho mayor. Este sería un punto muy destacable y favorable de Break Square sobre este juego.

El segundo juego (Dominó), ni siquiera tiene comunicación bluetooth lo que hace que el juego si no dispone de conexión a internet, sea completamente inútil en el modo de juego multijugador. Aun así, si dispusiese de conexión a internet y el usuario no dispone de Facebook, también es inútil el poder realizar una partida multijugador con amigos porque el Facebook es necesario para ello. Por último, añadir que, al estar tirando de datos móviles o wifi, al igual que el juego anterior, el consumo de batería es mucho mayor que en el caso de utilizar Bluetooth BLE.

Por último, decir, que ambos juegos (Tic Tac Toe y Domino) en su modo de juego multijugador, los jugadores son limitados, en Break Square pueden jugar a la vez tantos jugadores como quieran.

2.2 TEMÁTICA DE BREAK SQUARE Y JUEGOS SIMILARES

Como he explicado en apartados anteriores, Break Square consiste en parar los cuadros rojos que caen de la parte superior de la pantalla pulsando en ella, con el objetivo de dejarlos dentro de los cuadrados azules y eliminarlos intentando conseguir la puntuación más alta.

2.2.1 OTROS JUEGOS CON LA MISMA TEMÁTICA

Actualmente no he encontrado ningún juego con la misma temática que Break Square, que se de esa situación hoy en día es muy difícil por la cantidad de juegos que hay en Google Play, por tanto, eso también es un punto fuerte del juego.

Aunque no hay juegos parecidos, si es verdad que hay muchos juegos en el que la función principal para superarlos, se basa en ir pulsando continuamente la pantalla, a esta dinámica de juego se le llama TAP y no suelen tener modos multijugador o no son por turnos.

2.2.2 JUEGOS CON LA DINÁMICA DE JUEGO TAP

Actualmente hay muchos juegos con esta dinámica ya que son juegos simples y fáciles de jugar, que normalmente atraen a muchas personas ya que son muy adictivos. Uno de los ejemplos más conocidos es Flappy Bird, que fue el que puso de moda esta dinámica gracias a la simpleza del juego y el éxito que tuvo.

2.3 TECNOLOGÍA UTILIZADA EN EL PROYECTO.

Como he dicho en otros puntos anteriores, aparte de crear un juego con todos los detalles que podría tener cualquier juego realizado por una empresa, en este proyecto, quiero destacar el uso que se hace de la comunicación

bluetooth utilizando BLE (Bluetooth Low Energy) y haciendo uso de Beacons[1] (Balizas en español). Quiero destacar este apartado, ya que los Beacons no fueron creados para implementarlos en juegos, pero en algunos con unas características muy determinadas, pueden funcionar bastante bien.

Los Beacons fueron creados principalmente para implementarlos en el campo del marketing y en la geolocalización en interiores. Por este motivo, no hemos encontrado ningún juego que haga uso de esta tecnología, pero sí que hay bastantes aplicaciones que la utilizan, algunas de ellas son de empresas como: Coca Cola, Mc Donals, Nestle, KFC, Carlsberg, Air France, etc…

2.4 PÚBLICO AL QUE VA DIRIGIDO EL JUEGO

Teniendo en cuenta que es un juego casual, con una mecánica muy sencilla y de fácil uso, se podría decir que este juego va dirigido a toda aquella persona que se quiera entretener un rato (o varios) con él, sin límite de edad. En versiones anteriores lanzadas a la Play Store hay comentarios desde una madre que dice que a su hija pequeña le encanta, hasta personas mayores que dicen que no pueden parar de jugar porque es muy adictivo.

2.5 CONCLUSIÓN

Como conclusión de capítulo, he decir que también ha sido satisfactorio y espero que con el estudio de mercado que se ha hecho, haber resuelto las dudas ocasionadas y que pudiesen quedar del capítulo 1 con respecto al mismo y a lo que va enfocado dicho proyecto.

---

1 Un **beacon** es un dispositivo de bajo consumo que emite una señal broadcast, y son suficientemente pequeños para fijarse en una pared o mostradores. Utiliza conexión bluetooth de bajo consumo (BLE) para transmitir mensajes o avisos directamente a un dispositivo móvil sin necesidad de una sincronización de los aparatos, la señal es captada por estos dispositivos y se transmite a menudo a un servidor en la nube a través de internet. El servidor de la nube procesa la información y lleva a cabo análisis más detallado para guiar los comportamientos basados en la localización específica del dispositivo móvil. Fuente: https://es.wikipedia.org/wiki/Beacon

En el siguiente capítulo, me voy centrando en la parte técnica del proyecto en la que se hablará sobre el análisis y diseño del juego.

3 DEFINICIÓN DEL PROBLEMA

En este nuevo capítulo, se empieza con la parte técnica del proyecto donde empezaremos dando una visión acerca de la metodología y planificación llevada a cabo en el mismo, y, finalizaremos, definiremos los requisitos que deberá cumplir la aplicación una vez desarrollada.

3.1 METODOLOGÍA

Como se ha comentado en capítulos anteriores, en este proyecto, ya había elementos desarrollados, pero no estaban desarrollados aplicando los conocimientos adquiridos en la carrera, ya que, al empezar a desarrollar el juego, esos conocimientos no existían. Teníamos claro que habría que seguir una metodología de reestructuración del juego, así que empezamos haciendo un chequeo de lo que teníamos y a dónde queríamos llegar.

Por una parte, se replanteó el juego completo, empezamos a redefinir los requisitos, generamos los casos de uso y la estructura que tendría el juego.

Por otra parte, empezamos a estudiar las tecnologías y opciones que había para implementar las nuevas funcionalidades que tendría el juego, por

ejemplo, la comunicación. Y una vez que se tuvo claro la comunicación a utilizar, se creó un proyecto a parte, para ver si dicha comunicación podría cumplir los requisitos definidos. Comprobamos que podría valer y seguimos hacia delante.

Una vez finalizadas las 2 partes anteriores, se volvió a estudiar el código que ya teníamos, y, lo reutilizamos de acuerdo a los requisitos y casos de uso generados, siempre y cuando fuese posible y estuviese actualizado. Cuando no fue posible, generamos código nuevo y adaptamos el proyecto de la comunicación a nuestro juego.

Para finalizar, realizamos las pruebas necesarias para que el juego funcionase correctamente, y corregimos los posibles errores que pudieron salir.

Para todo el proceso anterior, llevamos a cabo una planificación.

3.2 PLANIFICACIÓN

Para llevar a cabo la parte de la metodología, se tuvo que cumplir en mayor o menor medida, con una planificación, dicha planificación nos serviría para llegar a tener el proyecto listo, y, ser presentado en la convocatoria de Enero-Febrero del año 2017.

Dicha planificación fue la siguiente:

1. En el mes de Agosto de 2016, se replantearía el juego completo empezando a definirse las funcionalidades, requisitos, casos de uso y la estructura que tendría dicho juego, por ejemplo, en los niveles, que no quedaban muy claros cómo implementarlos.
2. En el mes de Septiembre, se empezarían a estudiar las tecnologías adecuadas para implementar el modo de juego multijugador y se creó un proyecto nuevo para comprobar si la tecnología cumplía los requisitos.
3. En el mes de octubre, reutilizaríamos el código que ya teníamos y le aplicaríamos los conocimientos adquiridos en la carrera, como por ejemplo la POO (Programación Orientada a Objeto) con su herencia polimorfismo, sobrecarga de métodos, etc.

4. En el mes de noviembre, se añadiría el modo de juego en equipo adaptando al proyecto del juego, lo que ya teníamos desarrollado en el proyecto de la comunicación.
5. El mes de diciembre, se reservó para pruebas (la primera semana) y el tiempo restante se dejó para los posibles fallos que pudieran surgir de dichas pruebas y de margen de error, por si nos retrasábamos en algunos de los meses anteriores.
6. Por último, el mes de enero, se reservó, para generar la documentación.

3.3 REQUISITOS

Con los requisitos empezamos a definir nuestro juego, dichos requisitos nos irán indicando cuál será la funcionalidad de la aplicación y bajo qué circunstancias ésta va a funcionar de modo adecuado.

3.3.1 REQUISITOS FUNCIONALES

En este apartado se enumeran y describen los requisitos funcionales que tendrá el juego. Los requisitos funcionales son aquellos que especifican las funciones que tendrá el sistema (iteraciones con el sistema y su entorno), en este caso el juego.

TABLA 1: REQUISITOS FUNCIONALES

Identificador	Descripción
RF1	El juego será desarrollado para dispositivos móviles Android.
RF2	Se dispondrá de 3 modos de juego: Clásico, niveles, equipo.
RF3	Constará de leaderboards para que los usuarios puedan ver sus puntuaciones y puedan compararlas con la de otros jugadores.
RF4	Tendrá una sección de contacto para que el usuario pueda contactar con el creador a través de las redes sociales del juego o email.
RF5	Tendrá una sección de información en la que se explicará cómo funciona el juego.

RF6	Tendrá una opción para poder dar feedback al juego.
RF7	Incluirá publicidad entre las partidas de forma aleatoria máximo cada 3 partidas y mínimo cada 2.
RF8	Incluirá la opción de eliminar la publicidad previo pago.
RF9	Modo de juego clásico y modo de juego por equipos: los cuadrados superiores caerán en orden aleatorio y con velocidades aleatorias.
RF10	Modo de juego por niveles: para poder jugar a un nivel, será necesario haber superado el nivel anterior.
RF11	Al jugar una partida y finalizarla se dará la opción de compartir la puntuación a través de redes sociales, mensaje de texto, etc.
RF12	Será posible reiniciar una partida una vez finalizada.
RF13	Constará de dos leaderboard, uno será para el modo de juego por niveles y el otro para el modo de juego clásico.
RF14	Antes de empezar la primera partida en el juego, aparecerá un mensaje con las instrucciones para jugar al mismo.
RF15	Éste está relacionado con el RF14. Al mostrar el mensaje del RF14 se le dará la opción al usuario de no volver a mostrarlo
RF16	Modo de juego clásico: Los cuadrados pequeños y grandes se irán reactivando de forma aleatoria para que el juego no finalice mientras no falle el usuario.
RF17	Modo de juego clásico: Se informará al usuario cuando haya superado un record en su puntuación
RF18	Modo de juego en equipo: El creador de la partida le dará un nombre a la misma para que otros jugadores puedan identificarla y unirse a ella.
RF19	Modo de juego en equipo: Cuando un usuario decida unirse a una partida, verá una lista de las partidas disponibles y podrá seleccionar cualquiera de ellas.
RF20	Modo de juego en equipo: Los turnos de juego irán en el orden en el que los jugadores se hayan unido a la partida. El turno inicial siempre lo tendrá el creador de la partida.

RF21	Modo de juego en equipo: Todos los jugadores serán informados cuándo les toca el turno y cuándo les toca esperar.
RF22	Modo de juego en equipo: Todos los jugadores serán informados después de ser eliminado un cuadrado y de la puntuación total obtenida hasta el momento.
RF22	Modo de juego en equipo: Cuando termine la partida, se les notificará a todos los jugadores junto con la puntuación obtenida.
RF23	Modo de juego en equipo: Podrá unirse a la partida un número de usuarios ilimitado.

3.3.2 REQUISITOS NO FUNCIONALES

Los requisitos funcionales son los que afectan a los servicios o funciones del sistema, por tanto, delimitan las condiciones en que el sistema presta los servicios al usuario.

Dentro de los requisitos funcionales los vamos a dividir dependiendo de la categoría a la que pertenezcan.

3.3.2.1 RNF USABILIDAD

TABLA 2: RNF USABILIDAD

Requisitos	Descripción
RNFU1	El juego debe ser intuitivo a la hora de navegar por él.
RNFU2	El usuario sólo interactuará con el juego cuando sea estrictamente necesario.

3.3.2.2 RNF SOFTWARE

TABLA 3: RNF SOFTWARE

Requisitos	Descripción

RNFS1	El juego correrá en la plataforma Android a partir de la versión 3.0 (Honeycomb) incluida.

3.3.2.3 RNF HARDWARE

TABLA 4: RNF HARDWARE

Requisitos	Descripción
RNFH1	El dispositivo deberá integrar un chip bluetooth compatible con BLE (Bluetooth Low Energy)

3.3.2.4 RNF EN EL DISEÑO Y LA IMPLEMENTACIÓN

TABLA 5: RNF EN EL DISEÑO Y LA IMPLEMENTACIÓN

Requisitos	Descripción
RNFDI1	El juego se desarrollará en el lenguaje de programación JAVA.
RNFDI2	Se utilizará el SDK de Android para desarrollar el juego.
RNFDI3	El entorno de desarrollo (IDE) será Android Studio.

3.3.2.4 RNF RENDIMIENTO

TABLA 6: RNF RENDIMIENTO

Requisitos	Descripción
RNFR1	El uso de la batería será el mínimo posible.
RNFR2	El tiempo de respuesta del sistema debe ser aceptable.

3.4 CONCLUSIÓN

Como conclusión a este capítulo, he de decir que se ha realizado de forma exitosa y aunque en un principio nos retrasamos en el proyecto por motivos personales (yo, José Miguel, me fui a Reino unido), gracias a la planificación inicial, esos retrasos se pudieron solventar.

En el siguiente capítulo, continuaremos con los casos de uso y la estructura que tendrá el juego.

4 ANÁLISIS Y DISEÑO

En este capítulo, continuaremos con la parte referente a los casos de usos, diagrama de los elementos que intervienen en el juego y su explicación y el diseño anterior y actual de la interfaz del juego.

4.1 CASOS DE USO

En este apartado se declararán los casos de uso correspondientes al juego. Con ellos se definen de forma gráfica las acciones que pueden realizar los jugadores a la hora de interactuar con la aplicación.

Primeramente, se van a definir los actores que intervendrán y a continuación se expondrá el diagrama de casos de uso en los que participará dicho actor.

4.1.1 ACTORES

Los actores son una entidad externa que guardan relación con este juego y que demandan una funcionalidad. Sólo habrá un actor en el juego, el actor será el usuario o jugador.

4.1.2 DIAGRAMA Y DESCRIPCIÓN DE LOS CASOS DE USO

A continuación, se mostrará el diagrama de los casos de usos de los que consta el juego. Para facilitar la comprensión de los mismos, también se hará una descripción de cada uno de ellos.

ILUSTRACIÓN 14: CASOS DE USO

TABLA 7: CASO DE USO JUGAR PARTIDA

Nombre	*Jugar partida*
Descripción	Permite al usuario jugar una partida.
Precondición	Seleccionar un modo de juego.
Postcondición	El usuario ha jugado una partida.
Flujo principal	1- Se procesa el caso de uso “Empezar partida”. 2- Se procesa el caso de uso “Procesar partida”. 3- Se procesa el caso de uso “Terminar partida”.
Flujo alternativo	-

TABLA 8: CASO DE USO EMPEZAR PARTIDA

Nombre	*Empezar partida*
Descripción	Permite al usuario empezar una partida.
Precondición	Seleccionar un modo de juego.
Postcondición	El usuario ha empezado la partida.
Flujo principal	1- El juego informa al jugador de la mecánica del juego. 2- El juego queda a la espera de iniciar la partida cuando el usuario quiera.
Flujo alternativo	2b- El juego da el turno al jugador y comienza la partida.

TABLA 9: CASO DE USO PROCESAR PARTIDA

Nombre	*Procesar partida*
Descripción	El usuario está jugando la partida
Precondición	Se ha empezado una partida
Postcondición	Se ha procesado la partida
Flujo principal	1- Se inicia la bajada de un mini cuadrado. 2- El usuario pulsa la pantalla para parar el mini cuadrado y dejarlo dentro de un cuadrado. 3- El usuario elimina un cuadrado. 4- Se actualiza el contador de cuadrados. 5- Se vuelve al punto 1. 6- Se procesa el caso de uso “Terminar partida”.

Flujo alternativo	3b- El usuario no elimina un cuadrado. 6- Se procesa el caso de uso “Terminar partida”.

TABLA 10:CASO DE USO TERMINAR PARTIDA

Nombre	*Terminar partida*
Descripción	Termina la partida de un usuario.
Precondición	Se ha procesado una partida
Postcondición	Se ha terminado una partida.
Flujo principal	1- Aparece un mensaje con la puntuación obtenida y con la opción de compartir puntuación, reiniciar partida o salir. 2- Aparece un anuncio.
Flujo alternativo	2b- No aparece un anuncio.

TABLA 11: CASO DE USO SALIR PARTIDA

Nombre	*Salir partida*
Descripción	Permite al usuario salir de la partida.
Precondición	El usuario ha empezado una partida.
Postcondición	El usuario se ha salido de la partida.
Flujo principal	1- Se procesa el caso de uso “Procesar partida”. 2- Se procesa el caso de uso “Terminar partida”. 3- Se sale de la partida.
Flujo alternativo	1b- Se sale de la partida.

TABLA 12: CASO DE USO CREAR PARTIDA MULTIJUGADOR

Nombre	*Crear partida multijugador*
Descripción	Crea una partida multijugador
Precondición	Comprobar requisitos bluetooth y ser aptos
Postcondición	El usuario ha creado una partida multijugador
Flujo principal	1- Seleccionar crear partida. 2- Ejecutar caso de uso “Configurar crear partida”. 3- Esperar a que se unan jugadores.

Flujo alternativo	-

TABLA 13: CASO DE USO COMPROBAR REQUISITOS BLUETOOTH

Nombre	*Comprobar requisito bluetooth*
Descripción	Comprueba si se cumplen los requisitos bluetooth
Precondición	Seleccionar modo de juego en equipo o multijugador
Postcondición	Se han comprobado los requisitos bluetooth
Flujo principal	1- El sistema comprueba los requisitos bluetooth. 2- Cumple los requisitos bluetooth 3- Se ejecuta el caso de uso crear partida multijugador/unirse a partida multijugador según seleccione el usuario.
Flujo alternativo	2b- No cumple los requisitos bluetooth. 4- Vuelve a la vista de elegir modo de juego.

TABLA 14: CASO DE USO CONFIGURAR CREAR PARTIDA

Nombre	*Configurar crear partida*
Descripción	Configuración de la partida para crearla
Precondición	Seleccionar crear partida
Postcondición	Partida configurada.
Flujo principal	1- Rellenar campo nombre de la partida. 2- El sistema comprueba si está relleno el campo. 3- Si el campo está relleno la partida está configurada.
Flujo alternativo	3b- Si el campo no está relleno, se informa al usuario de que hay que rellenarlo.

TABLA 15: CASO DE USO REINICIAR PARTIDA

Nombre	*Reiniciar partida*
Descripción	Reinicia una partida
Precondición	Haber finalizado una partida.
Postcondición	Partida reiniciada.
Flujo principal	1- El usuario selecciona reiniciar partida.

	2- Se ejecuta el caso de uso jugar partida.
Flujo alternativo	-

TABLA 16: CASO DE USO UNIRSE A PARTIDA MULTIJUGADOR

Nombre	*Unirse a partida multijugador*
Descripción	El usuario se une a una partida multijugador o en equipo.
Precondición	Comprobar requisitos bluetooth y ser aptos.
Postcondición	El usuario se ha unido a la partida.
Flujo principal	1- El usuario selecciona unirse a partida 2- Ejecutar caso de uso configurar unirse a partida. 3- Ejecutar caso de uso seleccionar partida creada.
Flujo alternativo	-

TABLA 17: CASO DE USO CONFIGURAR UNIRSE A PARTIDA

Nombre	*Configurar unirse a partida*
Descripción	Se configura la partida para que el usuario se una a ella.
Precondición	Seleccionar unirse a partida.
Postcondición	El usuario se ha unido a la partida.
Flujo principal	1- Rellenar campo nombre de jugador. 2- El sistema comprueba si está relleno el campo. 3- Si el campo está relleno la partida está configurada.
Flujo alternativo	3a- Si el campo no está relleno, la partida no está configurada y se muestra un mensaje de que debes rellenarlo.

TABLA 18: CASO DE USO SELECCIONAR PARTIDA CREADA

Nombre	*Seleccionar partida creada*
Descripción	El usuario selecciona la partida creada a la que se unirá
Precondición	Debe estar configurado unirse a partida
Postcondición	El usuario ha seleccionado una partida
Flujo principal	1- El usuario ve las partidas creadas disponibles.

	2- El usuario selecciona la partida a la que se quiere unir entre todas la de la lista.
Flujo alternativo	-

TABLA 19: CASO DE USO VER INFORMACIÓN DE CONTACTO

Nombre	*Ver información contacto*
Descripción	El usuario verá una sección de contacto
Precondición	Estar en la primera vista del juego.
Postcondición	El usuario ha visto la sección de contacto.
Flujo principal	1- El usuario selecciona ver la sección de contacto. 2- El sistema abre una nueva vista con la información de contacto.
Flujo alternativo	-

TABLA 20: CASO DE USO VER FUNCIONAMIENTO DEL JUEGO

Nombre	*Ver funcionamiento del juego*
Descripción	El usuario verá información acerca del funcionamiento del juego
Precondición	-
Postcondición	El usuario habrá visto la información
Flujo principal	1- El usuario selecciona ver la información del funcionamiento del juego. 2- El sistema muestra la información en una ventana emergente.
Flujo alternativo	1b- El usuario inicia por primera vez el juego y en la primera partida, el sistema muestra una ventana emergente con la información del funcionamiento.

TABLA 21: CASO DE USO LOGEARSE EN GOOGLE PLAY GAMES

Nombre	*Logearse en Google Play Games*
Descripción	El usuario se logea con el juego en Google Play Games para disfrutar de más funcionalidades.

Precondición	No estar logeado con el juego en Google Play Games
Postcondición	El usuario se logea con el juego en Google Play Games.
Flujo principal	1- El usuario inicia el juego. 2- Le aparece una ventana emergente pidiéndole que se logee en Google play Games. 3- El usuario acepta y se logea.
Flujo alternativo	-

TABLA 22: CASO DE USO QUITAR PUBLICIDAD

Nombre	*Quitar publicidad*
Descripción	El usuario quita la publicidad que le aparece.
Precondición	El usuario ha jugado una partida.
Postcondición	El usuario ha quitado la publicidad.
Flujo principal	1- El sistema después de la partida, muestra publicidad. 2- El usuario en la parte superior selecciona la X para quitar dicha publicidad.
Flujo alternativo	2b- El usuario selecciona ir hacia atrás para quitar la publicidad.

TABLA 23:CASO DE USO VER LEADERBOARD

Nombre	*Ver leaderboard*
Descripción	El usuario ve el ranking del juego en los diferentes modos de juego.
Precondición	Estar en la vista principal del juego y estar logueado en google play games.
Postcondición	El usuario ha visto el leaderboard
Flujo principal	1- El usuario selecciona ver ranking. 2- Al usuario le aparece una ventana emergente para ver el ranking. 3- El usuario selecciona el leaderboard que desea (según el modo de juego) para ver el ranking
Flujo alternativo	-

TABLA 24: CASO DE USO COMPARTIR PUNTUACIÓN

Nombre	*Compartir puntuación*
Descripción	El usuario comparte información acerca de sus resultados en las redes sociales, por mensaje, etc.
Precondición	-
Postcondición	El usuario ha compartido su puntuación.
Flujo principal	1- El usuario se encuentra en la vista principal del juego. 2- El usuario selecciona compartir. 3- El usuario comparte el mensaje.
Flujo alternativo	1b- El usuario ha jugado una partida. 2- El usuario selecciona compartir. 3- El usaurio comparte el mensaje.

4.2 ESQUEMA GENÉRICO DE LA ESTRUCTURA DEL JUEGO

El siguiente esquema, no es un esquema formal, es más bien un esquema genérico con el conjunto de elementos más importantes del juego y cómo están relacionados entre sí. Con ello se pretende que se entienda mejor la estructura del juego sin que se sature el esquema con elementos innecesarios que no aportan ninguna información significante.

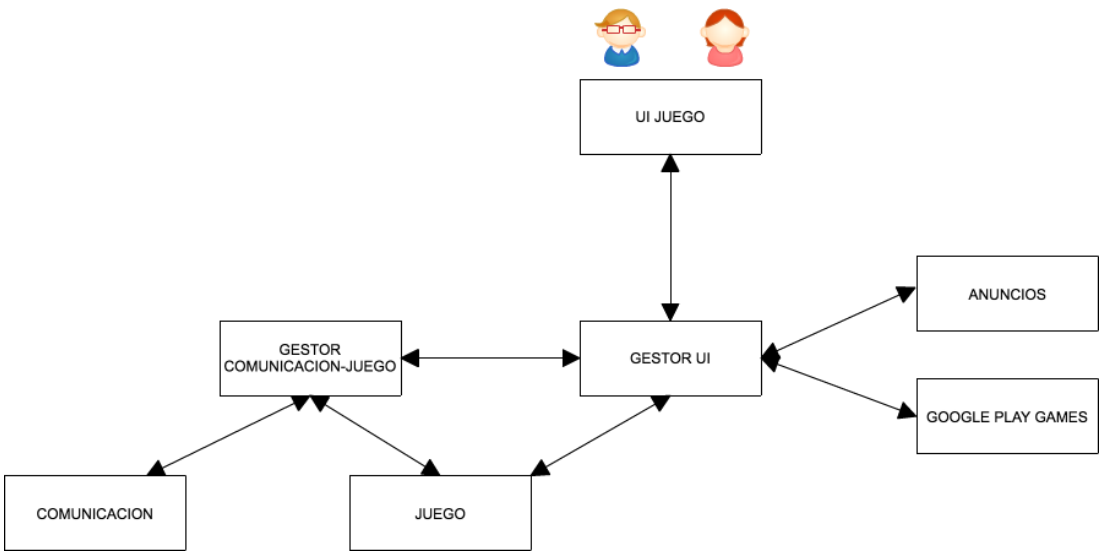

ILUSTRACIÓN 15: ESQUEMA GENÉRICO DE LA ESTRUCTURA DEL JUEGO

4.2.1 DESCRIPCIÓN DE LOS ELEMENTOS

En este sub-apartado voy a pasar a describir cada uno de los elementos que aparecen en la ilustración del punto anterior (ILUSTRACIÓN 14). Así quedará mucho más claro cómo está estructurado el juego.

4.2.1.1 UI JUEGO

Con este elemento me refiero a la parte gráfica del juego, es decir todo lo que va a aparecer en la pantalla y que el usuario va a poder ver. Por detrás se trabajará con otros elementos que serán totalmente invisibles al usuario.

Estará en permanente contacto con el GESTOR UI que será el encargado de ir transformando toda la información que reciba de otros elementos para procesarla y mostrarla al usuario.

4.2.1.2 GESTOR UI

Este elemento será el encargado de interpretar todos los datos recibidos por otros elementos, así como de gestionarlos para ofrecerlos en la interfaz del juego. Podría ser un Activity, un fragment, etc…

Es el elemento central y más grande el juego, por un lado, recibirá y enviará datos de ANUNCIOS, por otro al elemento GOOGLE PLAY GAMES y por otro (dependiendo del modo de juego seleccionado) al elemento GESTOR COMUNICACIÓN-JUEGO o directamente al elemento JUEGO.

4.2.1.3 ANUNCIOS

El elemento de anuncios será el encargado de hacer las peticiones de los anuncios que se vayan a mostrar en el juego, y que el GESTOR UI se encargará de mostrar.

Recibirá peticiones de anuncios del elemento GESTOR UI y le responderá a la petición una vez procesada.

4.2.1.4 GOOGLE PLAY GAMES

Será el encargado de gestionar el ranking de usuarios del juego que se basa en el servicio de juegos que ofrece Google para añadir a los desarrollos.

Por una parte, el GESTOR UI le enviará datos de usuarios, como las puntuaciones para irlas actualizando y por otra, el GESTOR UI y a petición del usuario, le pedirá que le pase el ranking de todos los usuarios del juego, este procesará dichas peticiones y le responderá según la petición.

4.2.1.5 GESTOR DE COMUNICACIÓN-JUEGO

Sólo se ejecutará cuando el modo de juego seleccionado sea por equipos y será el encargado de sincronizar lo que suceda en el juego con los datos enviados y recibidos para estar continuamente actualizado de lo que esté pasando en la partida en otros dispositivos.

El elemento de gestor de comunicación hace de observador y gestionará lo que va sucediendo entre el elemento de COMUNICACIÓN y el elemento JUEGO, hará de puente entre estos dos elementos cuando se quieran intercambiar información, pero además también estará pendiente de recibir y enviar peticiones al elemento GESTOR UI.

4.2.1.6 COMUNICACIÓN

Este elemento será el encargado de enviar y recibir mensajes en forma de beacons y actuar en consecuencia dependiendo de lo que envía y reciba.

Por una parte, gestionará los beacons entrantes, por ejemplo, de que un cuadrado se ha eliminado en otro dispositivo. El elemento COMUNICACIÓN recibirá ese mensaje, se lo comunicará al elemento GESTOR DE COMUNICACIÓN-JUEGO y este último, se lo comunicará al JUEGO, que eliminará el cuadrado que se ha eliminado en el dispositivo de otro jugador. Esto se aplica tanto a comunicaciones entrantes (una de ellas es el ejemplo que he puesto) como a comunicaciones salientes (cuando el dispositivo tiene que informar a otro de que el cuadrado se ha eliminado).

4.2.1.7 JUEGO

Es el juego en sí, la lógica del juego separada de toda la interfaz gráfica y de las comunicaciones existentes.

Aquí se gestionará todo lo que tiene que ver con el juego, desde los cuadrados que caen, hasta los cuadrados que se eliminan, pasando por la puntuación

obtenida y la detección de las colisiones (cuando un cuadrado pequeño, queda dentro de uno grande y se elimina).

4.2.2 EJEMPLO DE FLUJO SOBRE EL ESQUEMA

Una vez descritos todos los elementos que intervienen en el esquema y ver cuál es la funcionalidad de cada uno, vamos a ver un ejemplo sencillo, para ver la dirección que seguiría el flujo en una partida en el modo de juego “en equipo”, que es donde intervienen la mayoría de los elementos. Suponemos que la partida ya está configurada y los jugadores están esperando para empezar.

Inicio de la partida: El creador de la partida, es el que inicia la partida, para ello pulsa sobre la pantalla. Una vez que pulsa sobre la pantalla, GESTOR UI informa a GESTOR COMUNICACIÓN-JUEGO que se encarga de informar a JUEGO. JUEGO al recibir la información empieza la partida dejando caer el primer cuadrado de la parte superior de la pantalla. De forma paralela la parte gráfica se va actualizando con la posición del cuadrado en cada momento.

Parar cuadrado: Cuando el jugador vuelve a pulsar la pantalla, el objetivo es parar el cuadrado, así que al igual que ante, GESTOR UI informa a GESTOR COMUNICACIÓN-JUEGO que se encarga de informar a JUEGO. JUEGO en este caso, para el cuadrado cuando le llega la información y comprueba si dicho cuadrado ha quedado dentro de algún cuadrado grande. Suponemos que sí.

Enviar información a otros dispositivos: Una vez que se ha parado el cuadrado, y ha eliminado al cuadrado grande, el turno del jugador ha finalizado, por lo tanto, queda comunicar ambas cosas a los demás dispositivos (que un cuadrado grande ha sido eliminado y que el turno le pertenece a otro jugador). Para ello, JUEGO envía la información del cuadrado eliminado y de fin de turno a COMUNICACIÓN-JUEGO y este informa a COMUNICACIÓN (para avisar a los demás jugadores) y a GESTOR

UI para informar de que se ha terminado el turno. El juego seguiría hasta que una persona fallase.

Recibir información de otros dispositivos: Cuando un dispositivo está enviando, el otro está recibiendo y esto se gestiona todo de nuevo en el elemento COMUNICACIÓN que, en el caso de que un jugador haya fallado, recibirá el mensaje de fin de partida, se lo comunicará a GESTOR COMUNICACIÓN-JUEGO y este se lo notificará a JUEGO y a GESTOR UI para que realicen el procedimiento necesario, en este caso parar el JUEGO. Por otra parte, GESTOR UI al recibir la notificación de que se ha finalizado el juego, hará una solicitud al elemento ANUNCIOS para mostrar un anuncio una vez finalizada la partida.

Enviar puntuación y recibirla: Sólo en el modo de juego por niveles y clásico, entrará en juego el elemento GOOGLE PLAY GAMES y será una vez finalizada la partida, cuando JUEGO notifique a GESTOR UI que la partida ha finalizado y la puntuación y GESTOR UI la envie al elemento GOOGLE PLAY GAMES.

Por supuesto, debajo de todo lo anterior hay muchas funciones y comunicaciones más, pero se ha reducido al máximo para entender bien el flujo del ejemplo propuesto.

4.3 DISEÑO DE LA INTERFAZ

Primeramente, decir que en este aspecto no se nos ha formado en la carrera, y lo que he hecho, lo he hecho según mis gustos, preferencias e imitando el diseño de algunos juegos ya asentados en el mercado. Creo que el diseño ha quedado bien, aunque mejorable.

Para el diseño, se ha elegido un diseño limpio sin demasiados elementos, que sea intuitivo y en la misma línea en la que va el juego, una línea simple.

Al haber basado el diseño en mis gustos, no ha habido mock-ups reales, directamente he ido diseñando la interfaz a mi gusto y mejorándola a medida que se me iba ocurriendo e iba viendo en otras aplicaciones.

4.3.1 DISEÑO INICIAL DE LA PANTALLA PRINCIPAL VERSIÓN ANTERIOR

ILUSTRACIÓN 16: PRIMER DISEÑO DE LA PANTALLA PRINCIPAL DE BREAK SQUARE

En la imagen anterior se ve el diseño de lo que era la pantalla principal cuando el juego era simple y no tenía diferentes modos de juegos, sólo existía el actual modo clásico.

4.3.2 DISEÑO ACTUAL DEL JUEGO

En este apartado se dará a conocer las diferentes vistas de las que consta el juego en cada uno de sus estados.

La primera ilustración que vamos a ver se corresponde con la vista que tendríamos una vez iniciado el juego la primera vez. La ilustración de la izquierda es la que se verá para conectar con Google Play Games y la de la derecha, la vista principal del juego.

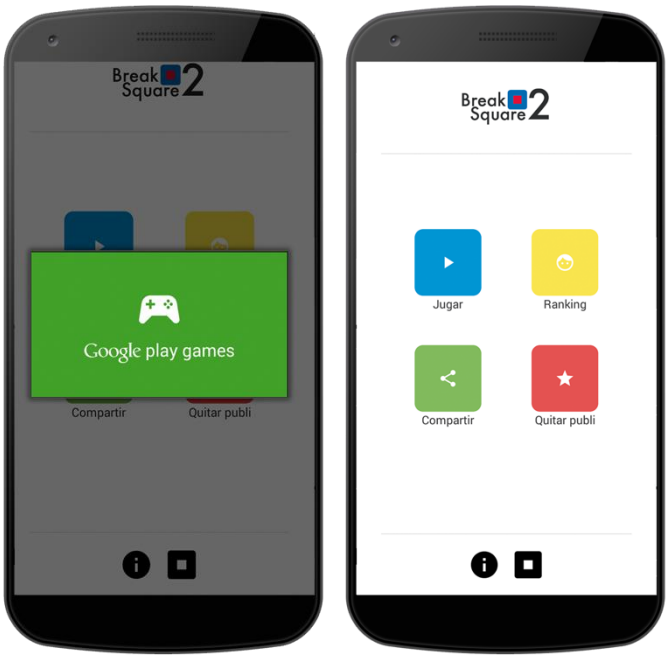

ILUSTRACIÓN 17: BREAK SQUARE: VISTA PANTALLA PRINCIPAL

En la siguiente ilustración vamos a ver qué se vería al pulsar sobre la información del juego:

ILUSTRACIÓN 18: BREAK SQUARE: VISTA DE INFORMACIÓN DEL JUEGO

Como se puede observar en la ilustración, se da información sobre el funcionamiento y los objetivos del juego.

A continuación, veremos la vista en el caso de que se pulse el botón de contacto:

ILUSTRACIÓN 19: BREAK SQUARE: VISTA CONTACTO

Como se puede comprobar en la ilustración anterior, se da la posibilidad al usuario de poder contactar conmigo en caso de alguna duda, sugerencia, bug, etc.

Seguidamente veremos la vista al pulsar sobre el ranking de los jugadores:

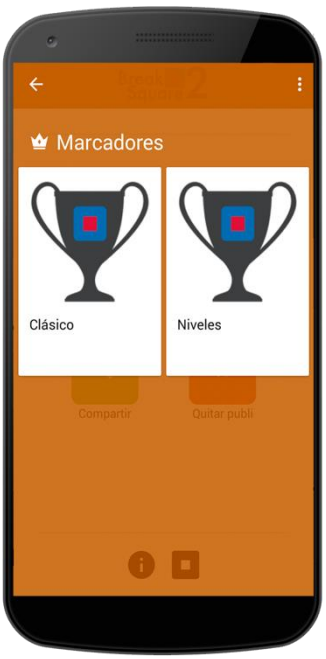

ILUSTRACIÓN 20: BREAK SQUARE: VISTA LEADERBOARD

En la vista anterior, aparecen 2 marcadores uno para el modo de juego clásico y el otro para el modo de juego por niveles, si seleccionamos uno de ellos, nos aparecerá un ranking de los jugadores con más puntuación en cada modo de juego.

A continuación, se verá la vista al pulsar la opción de compartir desde la pantalla principal.

ILUSTRACIÓN 21: BREAK SQUARE: VISTA AL PULSAR COMPARTIR

Desde la vista anterior, se podrá compartir el juego y la puntuación obtenida prácticamente a través de cualquier otra aplicación.

En la siguiente ilustración, se mostrará la vista cuando pulsamos sobre el botón jugar, en la que aparecen los diferentes modos de juego.

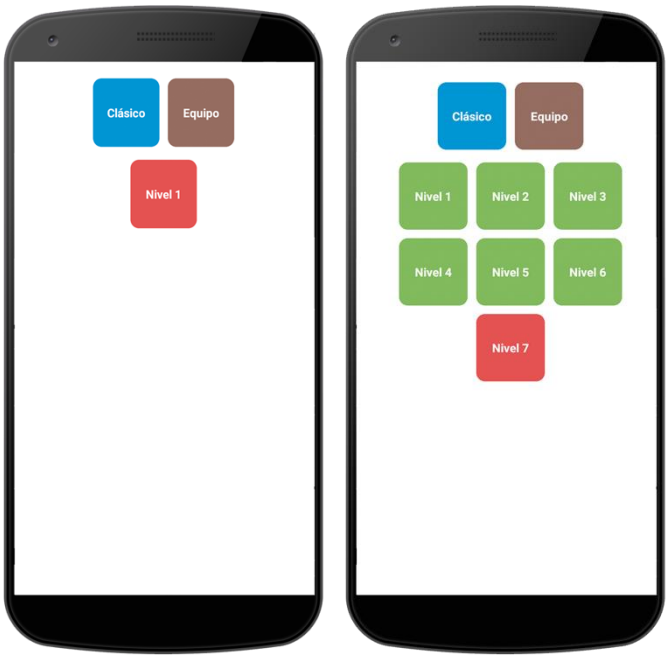

ILUSTRACIÓN 22: BREAK SQUARE: VISTA MODOS DE JUEGO

Como se puede comprobar en esta ilustración, aparecen los diferentes modos de juego (clásico, equipo y por niveles). Cuando se vayan superando niveles la vista será como la de la figura de la derecha.

En la siguiente ilustración, se verán las vistas del modo de juego clásico, la vista de un nivel y la vista del modo de juego en equipo.

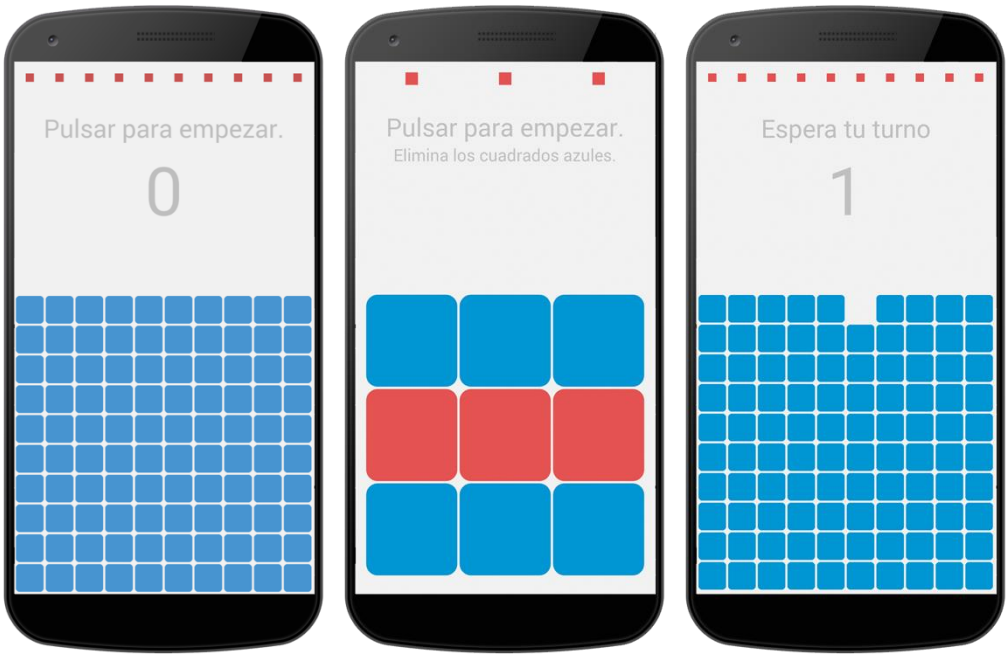

ILUSTRACIÓN 23: BREAK SQUARE: VISTA DIFERENTES MODOS DE JUEGO

La primera vista pertenece al modo de juego clásico, la segunda pertenece al modo de juego por niveles y por último, la tercera, pertenece al modo de juego en equipo.

En las siguientes imágenes, veremos el proceso de crear y unirse a una partida:

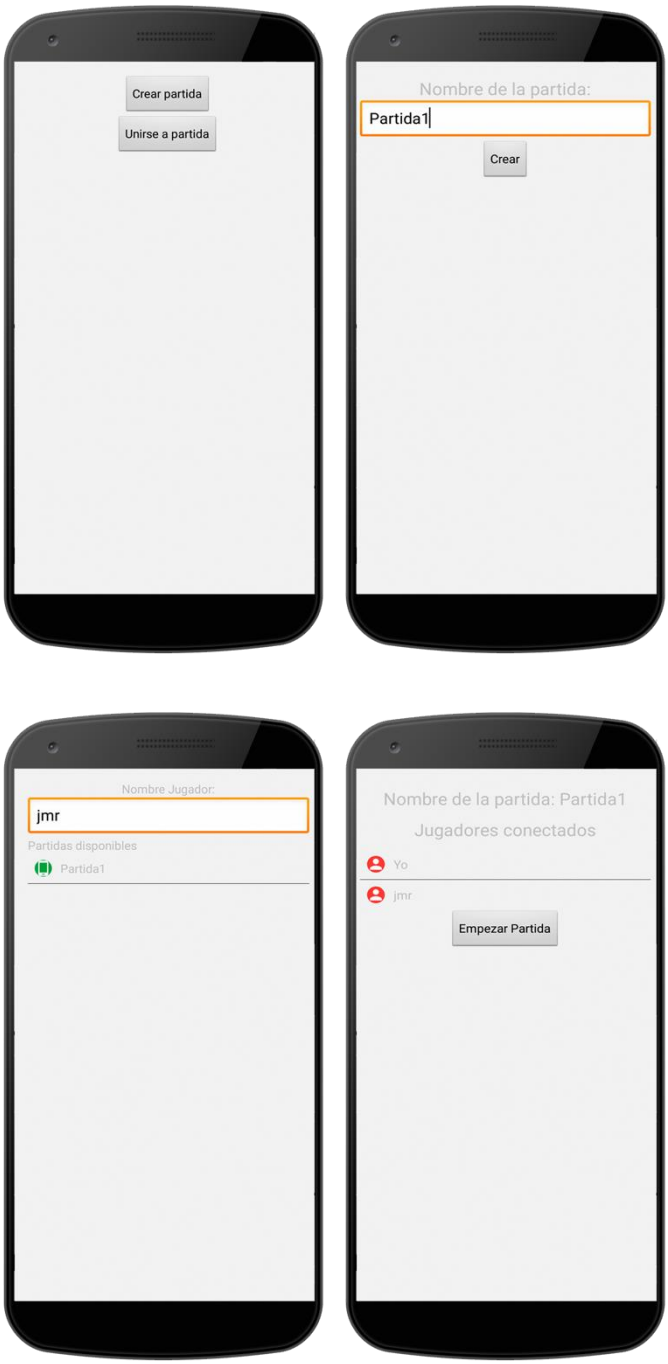

ILUSTRACIÓN 24: BREAK SQUARE: VISTAS PROCESO DE CREACIÓN DE PARTIDA EN MODO DE JUEGO “EQUIPO”

Como se puede observar, en la primera figura (arriba a la izquierda), una vez elegido el modo de juego en equipo, da la opción de crear una partida o unirse a ella.

En la segunda figura (arriba a la derecha), aparece la vista en caso de que el usuario hubiese dado a “crear partida”. En esta vista el usuario tiene que rellenar el nombre de la partida y pulsar el botón crear, para crearla.

En la tercera figura (abajo a la izquierda), aparece la vista en caso de que el usuario hubiese dado a “unirse a partida”. Lo que tendría que hacer el usuario esta vez, sería poner su nombre de usuario y seleccionar una partida pulsando sobre ella.

En la cuarta y última figura (abajo a la derecha), aparece la vista del creador de la partida cuando otro jugador se une a ella, en este caso cuando estuviesen todos los jugadores, el creador pulsaría el botón “EMPEZAR PARTIDA” y la partida daría comienzo.

4.4 CONCLUSIÓN

Como conclusión de este capítulo he de decir que también se ha superado de forma positiva, ya que se ha empezado definiendo los requisitos (tanto funcionales como no funcionales) de los que depende el juego y a continuación creando sus casos de usos y viendo la utilidad de los mismos, a continuación, se ha mostrado un esquema general del juego para ayudar a entender cómo van interconectados los elementos más importantes que intervienen y cuál es el flujo que sigue el juego en un ejemplo que se ha mostrado. Para finalizar con el capítulo se ha mostrado el entorno gráfico del que consta el juego.

En el próximo capítulo se hablará sobre cómo se ha llevado a cabo la implementación del juego y las tecnologías utilizadas, así como lo que se ha descubierto a medida que se iba implementando el jugo y conclusiones finales que se han sacado.

5 IMPLEMENTACIÓN

En este capítulo, se tratará todo lo relacionado con la implementación del proyecto, desde las tecnologías utilizadas a la hora de implementar el juego hasta los aspectos más relevantes del código fuente desarrollado.

En primer lugar, se mostrarán las tecnologías candidatas para abordar este proyecto y después con cada una de las seleccionadas para implementar en el proyecto se justificará el por qué han sido seleccionadas.

En segundo lugar, se indicarán las distintas librerías utilizadas para el proyecto indicando la utilidad de cada una de ellas.

En tercer lugar, me centraré en mostrar cómo se ha llevado a cabo la implementación de la comunicación para el modo de juego en equipo, así como el protocolo utilizado.

Y, para finalizar, me centraré en explicar el código fuente de las clases que he considerado más importantes a la hora de realizar este proyecto.

5.1 TECNOLOGÍAS CANDIDATAS Y UTILIZADAS

En este sub-apartado se hablará de las tecnologías candidatas y utilizadas para los distintos elementos que forman el juego. Dichos elementos son: El lenguaje o plataforma utilizada para desarrollar el juego, anuncios, leaderboad y comunicación.

5.1.1 PLATAFORMA UTILIZADA

Existen diferentes formas de realizar un juego tales como programar en lenguaje nativo en su plataforma, utilizar otra plataforma diferente como pueden ser Unity 3D[2], programar en lenguaje HTML con Apache Cordova[3], AngularJs[4], Ionic[5], etc.

En este proyecto se ha elegido programar en lenguaje nativo (Java) en la plataforma Android Studio por diferentes motivos:

- En la carrera hemos aprendido y estamos familiarizados con Java y la plataforma Android.
- Programar en lenguaje nativo tiene sus ventajas como, por ejemplo, tener un control más exhaustivo del código que se genera y, por consiguiente, restringir el uso que se hace de la batería, parte muy importante en este proyecto.

---

[2] **Unity 3D** es una de las **plataformas para desarrollar videojuegos más completas** que existen. Permite la creación de juegos para múltiples plataformas a partir de un único desarrollo, incluyendo el desarrollo de juegos para consola (PlayStation, Xbox y Wii), escritorio (Linux, PC y Mac), navegador, móviles y tabletas (iOS, Android, Windows Phone y BlackBerry).

[3] **Apache Cordova** es un framework para el desarrollo de aplicaciones móviles propiedad de Adobe Systems que permite a los programadores desarrollar aplicaciones para dispositivos móviles utilizando herramientas web genéricas como JavaScript, HTML5 y CSS3, resultando aplicaciones híbridas.

[4] **AngularJS** es un framework de JavaScript de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página.

[5] **Ionic** es una herramienta, gratuita y open source, para el desarrollo de aplicaciones híbridas basadas en HTML5, CSS y JS. Está construido con Sass y optimizado con AngularJS.

- El juego estaba empezado utilizando AndroidStudio y java por lo cual era mejor reutilizar el código antes que empezarlo de nuevo en otras plataformas desconocidas.
- El juego es bastante simple tanto en mecánica como en gráficos por tanto no hace falta un motor gráfico como Unity (previo pago de licencias si no queremos que nos aparezca publicidad sobre ello).

Como he comentado anteriormente, para desarrollar el proyecto, en este caso, he elegido programar en el lenguaje nativo de Android (Java) y en su entorno de desarrollo (Android Studio).

5.1.2 ANUNCIOS

A través de los anuncios obtendremos ingresos con nuestro juego, por lo cual es muy importante elegir el método que mejor se adapte a nuestras necesidades, ya que de eso dependerá que los ingresos en la aplicación sean superiores o no.

A la hora de elegir cómo poner los anuncios también hay varias posibilidades, desde hablar con empresas por cuenta propia proponiéndole mostrar sus anuncios en nuestro juego (conseguirlo es muy trabajoso y no siempre se consigue) hasta plataformas que se encargan de gestionarlo todo (la búsqueda de anunciantes y la demanda demanda de los mismos) y poner la publicidad más acorde dependiendo de la aplicación en cuestión. A ésta última posibilidad, pertenece la plataforma que ofrece Google llamada AdMob la cual he elegido para implementar en el proyecto.

He elegido AdMob por numerosos motivos, algunos de ellos son:

- AdMob es la plataforma más utilizada para estos fines ya que Google es una empresa puntera a la hora de demandar y ofrecer publicidad de todas las formas posibles.
- Al ser de Google y ser la más utilizada no nos vamos a tener que preocupar por si a corto plazo, se deja de utilizar o deja de dar soporte.
- Ofrece diferentes modos de publicidad, con los que poder ir probando cuál es el que más altos beneficios genera.

- Su implementación es bastante sencilla, por lo que se ahorra tiempo, tanto a la hora de implementarla como a la hora de modificarla.
- He utilizado ésta tecnología en otras ocasiones, estoy familiarizado con ella y puedo asegurar que funciona muy bien.

Como he comentado anteriormente, la tecnología elegida, en este caso, para implementar los anuncios en la aplicación, es AdMob.

5.1.3 LEADERBOARDS

Los leaderboards o tablas de clasificación, se utilizan para mostrar en los juegos cuáles son los usuarios que más puntuación consiguen y fomentar así la competitividad y que no se deje de jugar.

Para implementar leaderboads, hay diferentes posibilidades, desde montar un servidor que gestione todos los usuarios y puntuaciones (lleva mucho trabajo, difícil de mantener y costoso) hasta utilizar plataformas, que al igual que los anuncios, no lo gestionen todo y sólo tengamos que preocuparnos de enviar dichas puntuaciones. De esta última posibilidad hay dos opciones que sobresalen por encima del resto y que son las que aparecen en la mayoría de los juegos. Ellas son la plataforma de Google Play Games y la plataforma de Facebook Developers, que, entre otros servicios, ofrecen el de implementación y gestión de tablas de clasificación.

Normalmente se suele elegir una de ellas para implementarla en los juegos, pero podría ser también que por algún motivo se requiriesen los dos, lo cual, se puede hacer. En éste proyecto, como en la mayoría de juegos, he decidido utilizar sólo una. La elegida ha sido Google Play Games por una serie de motivos enumerados a continuación:

- Ambas son dos grandes multinacionales muy influyentes en estos temas, aunque destacaría a Google sobre Facebook porque su experiencia y recorrido es mayor con respecto a las aplicaciones.
- Este proyecto va enfocado a funcionar en el sistema operativo Android, por tanto, es necesario, que todo usuario que utilice Android, disponga de una cuenta de Google (necesaria para Google Play Games), sin

embargo, todo el mundo que utiliza Android, no necesita tener una cuenta de Facebook (también necesaria en caso de implementarlo en su plataforma). Por este motivo, he decidido utilizar Google Play Games.

- En ambas, la implementación es sencilla, aunque en Facebook tienes que realizar más validaciones y pasar más controles en caso de que lo implementes en su plataforma, lo que lleva más tiempo de desarrollo y no mejor servicio.
- Comparado con un servidor propio, he de decir que no tiene comparación ya que, con un servidor propio, tendríamos que gestionarlo todo, desde la creación de usuarios hasta el montaje y mantenimiento del servidor, lo que sería una tarea muy trabajosa y costosa, tanto en tiempo como en dinero.
- También he utilizado esta tecnología anteriormente y he comprobado que funciona muy bien.

Como he comentado anteriormente, la tecnología elegida en este caso, para implementar los leaderboards, es Google Play Games.

5.1.4 COMUNICACIÓN

En éste sub-apartado, se hablará de la tecnología utilizada para llevar a cabo el proceso de comunicación, necesario para el modo de juego en equipo. También hablaré sobre los requisitos que tiene que cumplir un juego determinado para poder implementar la tecnología elegida, ya que, a raíz del proyecto, he descubierto que dicha tecnología no es válida para todos los juegos, sólo para aquellos, que cumplen dichos requisitos.

Para realizar la comunicación para el modo de juego multijugador se puede hacer de varias formas, a continuación, las numero y después paso a explicar por qué elijo una y descarto las demás:

1- Bien montando un servidor que gestione las comunicaciones a través de internet y todo lo que ello conlleva como el mantenimiento, la gestión de usuarios, etc.

2- Realizar una comunicación a través de Google Play Services que al fin y al cabo es lo mismo que lo del punto 1, ya que el servidor lo necesitas, pero te lo facilita Google y además también te facilita el mantenimiento y la gestión de usuarios ya que al igual que en los leaderboards, depende de ellos.
3- Realizar una comunicación vía wifi como hacen muchos juegos y que es una práctica cada vez más extendida.
4- Realizar una comunicación vía bluetooth, que al igual que la anterior, también es muy utilizada en los juegos.

A continuación, describo los motivos que me han llevado a elegir una tecnología de las anteriores.

El primer punto (servidor propio) y el segundo (Google Play Games) los descarto automáticamente ya que uno de los requisitos principales del juego es que se pueda jugar a todas sus modalidades de forma offline (sin necesidad de internet). Por tanto, los dos primeros puntos descartados por necesitar internet para implementarlos.

Por consiguiente, debemos seleccionar la mejor opción de las que nos quedan (wifi o bluetooth), dependiendo de los requisitos del juego. Por una parte, vamos a explicar ambas tecnologías y a continuación se dará una conclusión de la que mejor se adapta al proyecto.

5.1.4.1 WIFI

El **wifi** es un mecanismo de conexión de dispositivos electrónicos de forma inalámbrica. Los dispositivos habilitados con wifi (como una computadora personal, un televisor inteligente, una videoconsola, un teléfono inteligente o un reproductor de música) pueden conectarse a internet a través de un punto de acceso de red inalámbrica. Dicho punto de acceso tiene un alcance de unos veinte metros en interiores, distancia que es mayor al aire libre. Fuente: https://es.wikipedia.org/wiki/Wifi

Puede decirse que WIFI es una de las tecnologías inalámbricas más utilizadas y extendidas desde su aparición. Hoy en día muchos aparatos electrónicos la tienen implementada, entre ellos, todos los ordenadores, tablets, teléfonos

móviles y muchas videoconsolas que usamos, y, además, se está haciendo popular en los electrodomésticos gracias al IoT (Internet de las cosas).

Actualmente hay varios estándares wifi, todos ellos parten del estándar inicial 802.11. La diferencia de los distintos estándares radica en diferentes características como la frecuencia usada, el ancho de banda, la velocidad, el alcance, etc. A continuación, se muestra una tabla con dichos estándares.

TABLA 25: ESTÁNDARES WIFI

FUENTE: HTTPS://NORFIPC.COM/REDES/TIPOS-REDES-ESTANDARES-WI-FI-DIFERENCIAS.PHP

Estandar	Velocidad (teórica)	Velocidad (práctica)	Frecuencia	Ancho de banda	Alcance (metros)	Detalles	Año
802.11	2 Mbit/s	1 Mbit/s	2,4 Ghz	22 MHz	330		1997
802.11a	54 Mbit/s	22 Mbit/s	5,4 Ghz	20 MHz	390		1999
802.11b	11 Mbit/s	6 Mbit/s	2,4 Ghz	22 MHz	460		1999
802.11g	54 Mbit/s	22 Mbit/s	2,4 Ghz	20 MHz	460		2003
802.11n	600 Mbit/s	100 Mbit/s	2,4 Ghz y 5,4 Ghz	20/40 MHz	820	Disponible en la mayoria de los dispositivos modernos. Puede configurarse para usar solo 20 MHz de ancho y asi prevenir interferencias en una zona congestionada	2009
802.11ac	6.93 Gbps	100 Mbit/s	5,4 Ghz	80 o hasta 160 MHz		Nuevo estándar sin interferencia pero con menos alcance, aunque hay tecnologías que lo amplían. Más rendimiento y otras ventajas.	2013
802.11ad	7.13 Gbit/s	Hasta 6 Gbit/s	60 Ghz	2 MHz	300		2012
802.11ah			0.9 Ghz		1000	Wi-Fi HaLow	2016

5.1.4.1.1 WIFI-DIRECT

Wi-Fi Direct es una norma que permite que varios dispositivos Wi-Fi se conecten entre sí sin necesidad de un punto de acceso intermedio. Fuente: https://es.wikipedia.org/wiki/Wi-Fi_Direct

Esta tecnología podría ser implementada en nuestro Proyecto ya que permite la comunicación entre dos dispositivos.

5.1.4.2 BLUETOOTH

Bluetooth es una especificación industrial para Redes Inalámbricas de Área Personal (WPAN) que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia en la banda ISM de los 2.4 GHz. Fuente: https://es.wikipedia.org/wiki/Bluetooth

También hay diferentes estándares Bluetooth al igual que en la wifi y hay una nueva tecnología llamada Bluetooth Low Energy que hace que se reduzca el consumo de energía del dispositivo que la implementa de forma notable.

5.1.4.2.1 BLUETOOTH LOW ENERGY (BLE)

Bluetooth de baja energía, también denominada como Bluetooth LE, Bluetooth ULP (Ultra Low Power) y Bluetooth Smart, es una nueva tecnología digital de radio (inalámbrica) interoperable para pequeños dispositivos desarrollada por Bluetooth. Fuente: https://es.wikipedia.org/wiki/Bluetooth_de_baja_energía

BLE nace para el internet de las cosas, a diferencia del Bluetooth que permite transmitir grandes volúmenes de datos, este no lo permite, pero en cambio se ve afectada de forma positiva el uso que hace de la batería, que es mínimo con respecto al Bluetooth clásico.

Por tanto, BLE junto a Wifi-direct son las dos candidatas finales con opciones a ser implementadas en el juego.

5.1.4.3 BLUETOOTH LOW ENERGY VS WIFI-DIRECT

Teniendo en cuenta toda la información del apartado anterior, se ha elegido la tecnología Bluetooth Low Energy para ser implementada en el proyecto, por los siguientes motivos:

- Consume poca energía y con ello se consigue uno de los objetivos principales de la aplicación, consumir el mínimo de batería posible.
- El tamaño de datos enviados que necesita el juego para llevar a cabo la comunicación, es bajo por lo que caben en los mensajes de los beacons a lanzar.
- Es una tecnología inalámbrica reciente, lo que hace que se cumpla otro objetivo del proyecto.
- El dispositivo no necesita estar conectado a internet.

5.1.4.4 CARACTERÍSTICAS QUE DEBE CUMPLIR UN JUEGO PARA IMPLEMENTAR BLE

A medida que se ha ido desarrollando el juego e implementando BLE, se han descubierto dos requisitos imprescindibles que un juego debería cumplir para poder implementar BLE en el modo multijugador. A continuación, explico cada uno de ellos y propongo una solución:

1- Si en el juego se necesita envío bidireccional constante de datos, esta tecnología no puede ser usada, ya que es demasiado lenta para enviar mensajes bidireccionales constantemente y los mensajes que llegan, pueden no llegar en orden. En este caso se tendría que utilizar wifi-direct.

2- Si el juego cumple lo anterior, pero necesita enviar muchos datos en un mensaje, tampoco sería posible ya que una de las características para ahorrar energía es que los mensajes enviados ocupan menos. En este caso, de nuevo habría que utilizar wifi-direct.

5.2 LIBRERÍAS

En este apartado se nombrarán las librerías utilizadas para llevar a cabo el proyecto y se describirán para facilitar la comprensión de las mismas.

5.2.1 ALTBEACON

AltBeacon es una librería que fue creada para interactuar con beacons, ya sean iBeacons (Apple), Eddystone (Google) o AltBeacon (Radious Network).

Estos últimos vienen a ser la respuesta a los iBeacons ya que cubren las mismas funcionalidades, que iBeacon pueden ofrecer. Aunque su apoyo no es tan amplio, ya hay grandes empresas que están utilizando esta tecnología. Algunas de ellas son; Coca-cola, Mc Donals, KFC, AirFrace, Carlsberg, Mango, Nestle, etc.

Esta tecnología permite tanto detectar beacons, como transformar el dispositivo (en este caso móvil o Tablet) en emisor de beacons.

Cualquier dispositivo, con una versión de Android mayor a la 4.3 y chipset Bluetooth Low Energy, puede detectar beacons. En junio de 2015, según datos de Google Play, eran el 56% de dispositivos móviles Android. Sin embargo, para poder transmitir, se necesita una versión superior a la 5.0 de Android y firmware soportando Bluetooth Low Energy.

5.2.1.1 ESPECIFICACIÓN

AltBeacon es una especificación de protocolo que define un formato de mensaje para anunciar beacons por proximidad. El contenido del mensaje contiene información que el receptor puede utilizar para identificar la baliza y calcular su distancia relativa a dicho beacon. El dispositivo receptor también puede utilizar la información del mensaje para ejecutar procedimientos y comportamientos, según su contenido.

5.2.1.1.1 OBJETIVOS DEL DISEÑO

1- Proporcionar mensajes por proximidad y poder intercambiar información entre emisores y receptores.
2- Mantener el cumplimiento con la especificación Bluetooth versión 4.0.
3- Facilitar la adopción de esta tecnología a todas las partes interesadas sin restricciones obvias de la aplicación.
4- Habilitar la implementación de características específicas del proveedor si es posible.

5.2.1.1.2 FORMATO DEL PROTOCOLO

AltBeacon hace uso de la estructura de datos para emisión definida en la especificación Bluetooth 4.0 (Volume 6, Part B, Section 2.3 Advertising Channel PDU).

La unidad de datos de la especificación AltBeacon está dividida en varios campos y cada uno de ellos, tiene una función en concreto. En la siguiente ilustración podemos ver la unidad de datos de la especificación AltBeacon que cuelga de la unidad de datos de la especificación BLE:

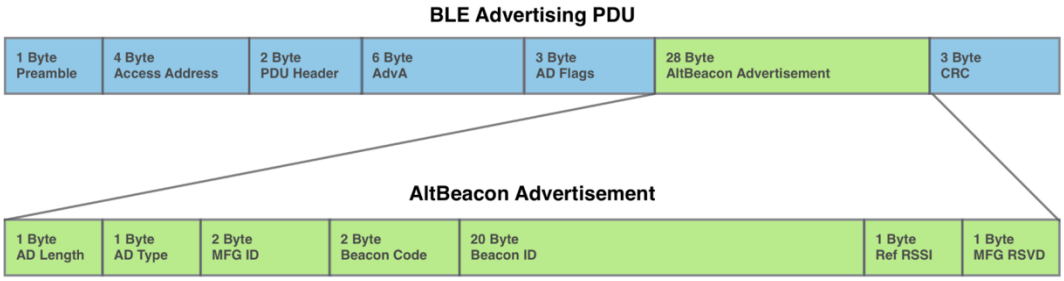

ILUSTRACIÓN 25: UNIDAD DE DATOS DE ALTBEACON

FUENTE: HTTPS://GITHUB.COM/ALTBEACON/SPEC

Como se puede comprobar en la imagen anterior, hay diferentes campos y cada uno tiene su función, en la siguiente ilustración se muestra la función de cada uno y los valores aceptados:

Field Name	Description	Accepted Values
AD LENGTH [MFG SPECIFIC]	Length of the type and data portion of the Manufacturer Specific advertising data structure.	`0x1B`
AD TYPE [MFG SPECIFIC]	Type representing the Manufacturer Specific advertising data structure.	`0xFF`
MFG ID	The beacon device manufacturer's company identifier code.	The little endian representation of the beacon device manufacturer's company code as maintained by the Bluetooth SIG assigned numbers database
BEACON CODE	The AltBeacon advertisement code	The big endian representation of the value `0xBEAC`
BEACON ID	A 20-byte value uniquely identifying the beacon	The big endian representation of the beacon identifier. For interoperability purposes, the first 16+ bytes of the beacon identifier should be unique to the advertiser's organizational unit. Any remaining bytes of the beacon identifier may be subdivided as needed for the use case.
REFERENCE RSSI	A 1-byte value representing the average received signal strength at 1m from the advertiser	A signed 1-byte value from 0 to -127
MFG RESERVED	Reserved for use by the manufacturer to implement special features	A 1-byte value from `0x00` to `0xFF`. Interpretation of this value is to be defined by the manufacturer and is to be evaluated based on the `MFG ID` value

ILUSTRACIÓN 26: DESCRIPCIÓN CAMPOS PROTOCOLO ALTBEACON

FUENTE: HTTPS://GITHUB.COM/ALTBEACON/SPEC

De los campos anteriores, nos vamos a centrar en el campo BEACON ID, que como dice la descripción, los 20 bytes son para la identificación del beacon y para utilizarlos según necesitemos dependiendo del uso. Los 20 bytes están subdivididos en 3 campos; UUID (16 byte) + Major ID (2 byte) + Minor ID (2 byte) que serán los que utilicemos para implementar nuestro protocolo de comunicación.

5.2.1.2 INSTALACIÓN EN ANDROID

Su implementación es muy sencilla ya que en Android Studio se utiliza por defecto el gestor de dependencias Gradle, lo único que tenemos que hacer es añadir la siguiente línea a nuestro archivo **build.gradle** del proyecto concretamente en la sección **dependencies.**

```
dependencies {
...
    compile 'org.altbeacon:android-beacon-library:2+'
...
}
```

Como se puede observar en la línea de implementación de la librería, estamos usando la versión 2 de dicha librería.

5.2.2 GOOGLE PLAY SERVICE

Es una API, que nos ofreces distintos y variados servicios, como pueden ser Google maps, Google Drive, Google Cloud, Google play games services, Android pay, Google Mobile Ads, etc…

De los servicios anteriores nosotros en nuestro juego hemos implementado Google play games services necesario para nuestros leaderboards y Google Mobile Ads necesario para mostrar los anuncios que aparecen en la aplicación.

Google, también nos ofrece una librería llamada **BaseGamesUtils** para simplificar nuestro código a la hora de gestionar los errores de conexiones y la visualización en los diálogos de dichos errores.

5.2.2.1 INSTALACIÓN

Instalar la librería BaseGamesUtils es muy sencillo. A continuación, se explica paso por paso el procedimiento.

1- Primero, tenemos que descargarla, y para ello, lo haremos de la páginas de ejemplos de Android (ir a la página de ejemplos de Android).
2- Al estar trabajando con Android Studio, la importaremos a nuestro proyecto yendo a **File > Import Module** y donde tenemos descargada la librería, seleccionamos **BaseGamesUtils** que está dentro de **BasicSamples/libraries**.
3- Por último se tiene que añadir a nuestro build.gradle (sección **dependencies**) del proyecto para que se compile, añadiendo lo siguiente:

```
dependencies {
...
    compile project(path: ':BaseGameUtils')
...
}
```

4- Por último, tendríamos que importar en nuestras clases del proyecto en las que fuese a ser utilizado, tanto la librería como Google Play Services.

```
import com.google.android.gms.*;
import com.google.example.games.basegameutils.BaseGameUtils;
```

No haría falta compilar también Google Play Services, ya que está compilado en la librería que hemos añadido y eso nos es de utilidad para todo el proyecto.

5.2.3 IN-APP BILLING

Este servicio tiene como objetivo, enviar solicitudes de facturación integrada y gestionar transacciones de facturación integrada a través de Google Play. En este proyecto la vamos a utilizar para que cuando el usuario pague un precio determinado, no vuelva a aparecer más publicidad, pero se podría usar para muchas cosas más, tales como subscripciones, compras integradas (como cuando se compra un elemento determinado en un juego) o todo lo que se nos pueda ocurrir que se pueda vender.

5.2.3.1 INSTALACIÓN

La instalación de esta librería también es muy sencilla y hay varios métodos, voy a explicar el que he realizado yo. A continuación, se explica paso por paso el procedimiento para ello:

A. Descargar la librería, para ello descargar el archivo IInAppBillingService.aidl (descarga).
B. Dirígete a **src/main** del proyecto.
C. Selecciona **File > New > Directory** y ponle de nombre al directorio **“aidl”**.
D. Selecciona **File > New > Package** y ponle de nombre **“com.android.vending.billing”**
E. Copia el archivo descargado en el punto **A** y pégalo dentro del paquete.

F. El resultado tiene que quedar como en la siguiente imagen:

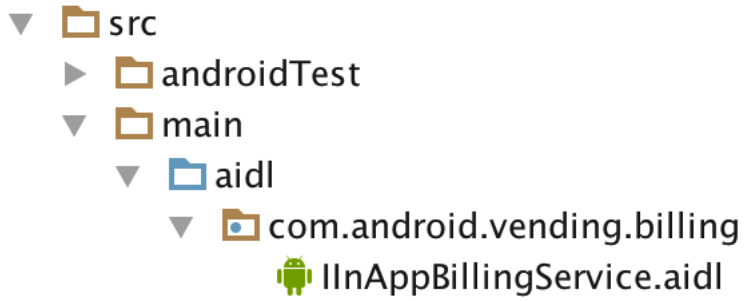

ILUSTRACIÓN 27: VISTA INSTALACIÓN LIBRERÍA IINAPPBILLINGSERVICE

G. A continuación, hay que actualizar el manifiesto de la aplicación (AndroidManifiest.xml) añadiendo la siguiente línea:

```
<uses-permission android:name="com.android.vending.BILLING" />
```

H. La librería ya está lista para poder ser utilizada en nuestro proyecto.

5.3 COMUNICACIÓN ENTRE DISPOSITIVOS

En este apartado, se profundizará sobre el protocolo creado para que se produzca una comunicación efectiva entre los dispositivos participantes en el modo de juego en equipo.

5.3.1 PROTOCOLO

Para que la comunicación entre dispositivos sea adecuada y ordenada se ha creado un protocolo en la que los dispositivos dependiendo del rol que jueguen o del mensaje que les llegue, puedan actuar y responder de una forma u otra. Dicho protocolo es muy sencillo y fácil de entender.

Como se ha comentado en uno de los apartados anteriores (ver formato del protocolo AltBeacon) para implementar nuestro protocolo, vamos a utilizar el campo BEACON ID (formado por los campos UIID(id1), Mayor ID(id2), Minor Id(id3)) para transmitir los mensajes que necesitamos.

5.3.1.1 MECANISMO

El mecanismo que vamos a utilizar en este protocolo, es el mecanismo de paso de mensajes. A continuación, se explicarán cada uno de estos mensajes, indicando también quién los envía, si el dispositivo que crea la partida o el dispositivo que se une a ella. Dichos mensajes serán enviados a través de beacons.

- El primer campo (UUID o id1) es el más amplio de los 3 comentados anteriormente, está formado por 16 bytes. Este campo lo utilizaremos para enviar la información necesaria a otros dispositivos, tales como dirección del receptor, dirección del receptor o datos de la partida.
- El segundo campo (Mayor id o id2) tiene una amplitud de 2 bytes identificará el tipo de mensaje que estamos enviando. Los posibles mensajes son los siguientes:
    - BEACON_NUEVA_PARTIDA: Este mensaje será enviado por el dispositivo que crea la partida, cuando se creé esta misma. Dicho mensaje informa de que hay una partida disponible para poder unirse a ella.
    - BEACON_UNIRSE_PARTIDA: Este mensaje será enviado por el dispositivo que se quiera unir a una partida existente. Dicho mensaje informa de que el usuario quiere unirse a una partida concreta.
    - BEACON_TU_TURNO: Este mensaje será enviado por el dispositivo creador de la partida, para informar al destinatario de que le ha llegado el turno de partida.
    - BEACON_FIN_TURNO: Este mensaje será enviado por el dispositivo que se une a la partida y que tiene el turno. Cuando finalice dicho turno, enviará este mensaje para informar al creador de la partida del cuadrado eliminado y de que su turno ha finalizado, pudiendo el creador, asignar el turno a otro dispositivo.
    - BEACON_ULTIMO_TURNO: Este mensaje será enviado por el dispositivo que crea la partida par informar a los que están unidos a ella del cuadrado que ha eliminado el jugador que ha tenido el último turno.

  - BEACON_FIN_PARTIDA: Este mensaje será enviado por el dispositivo creador de la partida cuando un jugador haya fallado, para informar de que ha finalizado la partida.
- El tercer campo (Minor Id o id3) se utilizará para identificar la partida en cuestión a la que se está jugando. Éste campo contendrá un número del 0 al 9.999.

Según lo comentado anteriormente, el tercer campo siempre contiene lo mismo en la partida, sin embargo, el Id1 es diferente por cada Id2 posible, a continuación, se explica cómo está estructurado el Id1 dependiendo del Id2:

- BEACON_NUEVA_PARTIDA:
  - Dirección del creador (C): Dirección Bluetooth del dispositivo creador de la partida. Ocupa 6 Bytes (12 en hexadecimal).
  - Nombre de la partida (P): Nombre que tendrá la partida. Ocupa 10 Bytes (20 en hexadecimal).
  - Formato:

    0xCCCCCCCCCCCCPPPPPPPPPPPPPPPPPPPP

- BEACON_UNIRSE_PARTIDA:
  - Dirección del creador (C): Dirección Bluetooth del dispositivo creador de la partida. Ocupa 6 Bytes (12 en hexadecimal).
  - Dirección del unido (U): Dirección Bluetooth del dispositivo unido a la partida. Ocupa 6 Bytes (12 en hexadecimal).
  - Nombre del jugador (N): Nombre del jugador que se une a la partida. Ocupa 4 Bytes (8 en hexadecimal).
  - Formato:

    0xCCCCCCCCCCCCUUUUUUUUUUUUNNNNNNNN

- BEACON_TU_TURNO:
  - Dirección del creador (C): Dirección Bluetooth del dispositivo creador de la partida. Ocupa 6 Bytes (12 en hexadecimal).
  - Dirección del turno (T): Dirección Bluetooth del dispositivo al que se le va a ceder el turno. Ocupa 6 Bytes (12 en hexadecimal).

  - Carácter blanco en hexadecimal (7F) para los 4 Bytes restantes (8 en hexadecimal)
  - Formato:

0xCCCCCCCCCCCCTTTTTTTTTTTT7F7F7F7F

- BEACON_FIN_TURNO:
  - Dirección del creador (C): Dirección Bluetooth del dispositivo creador de la partida. Ocupa 6 Bytes (12 en hexadecimal).
  - Dirección del turno (T): Dirección Bluetooth del dispositivo que ha finalizado el turno. Ocupa 6 Bytes (12 en hexadecimal).
  - Carácter blanco en hexadecimal (7F) para los 3 Bytes restantes (6 en hexadecimal).
  - Número del Big Square eliminado (N): Número del Big Square que ha sido eliminado en el turno, se utiliza para informar al creador de la partida de que ha sido eliminado y que avise a los otros jugadores de la partida. Si en el turno no se ha eliminado ningún Big Square el número será -1. Ocupa 1 Byte (2 en hexadecimal),
  - Formato:

0xCCCCCCCCCCCCTTTTTTTTTTTT7F7F7FNN

- BEACON_ULTIMO_TURNO:
  - Dirección del creador (C): Dirección Bluetooth del dispositivo creador de la partida. Ocupa 6 Bytes (12 en hexadecimal).
  - Dirección del último turno (T): Dirección Bluetooth del dispositivo que ha tenido el último turno. Ocupa 6 Bytes (12 en hexadecimal).
  - Carácter blanco en hexadecimal (7F) para los 3 Bytes restantes (6 en hexadecimal).
  - Número del Big Square eliminado (N): Número del Big Square que ha sido eliminado en el último turno, se utiliza para informar a los jugadores de la partida de que deben eliminar el número del Big Square indicado. Ocupa 1 Byte (2 en hexadecimal),
  - Formato:

0xCCCCCCCCCCCCTTTTTTTTTTTT7F7F7FNN

- BEACON_FIN_PARTIDA:
  - Dirección del creador (C): Dirección Bluetooth del dispositivo creador de la partida. Ocupa 6 Bytes (12 en hexadecimal).
  - Carácter blanco en hexadecimal (7F) para los 10 Bytes restantes (20 en hexadecimal).
  - Formato:

    0xCCCCCCCCCCCC7F7F7F7F7F7F7F7F7F7F

5.4 ENTORNO DE DESARROLLO

En este apartado se recogen en hardware y las herramientas para llevar a cabo la realización del proyecto.

5.4.1 HARDWARE UTILIZADO PARA EL DESARROLLO

Para llevar a cabo el desarrollo de la aplicación, se ha hecho uso de un ordenador portátil con características suficientes para hacer funcionar las herramientas software utilizadas. Las características son:

- MacBook Pro
  - CPU: Intel Core i7 2,5 GHz.
  - Memoria Ram: 16 GB 1600 MHz DDR3.
  - Tarjeta gráfica: NVIDIA GeForce GT 750M.
  - HDD: 500 GB.
  - Sistema operativo: macOS.

Para realizar pruebas y comprobar el correcto funcionamiento de la aplicación se han utilizado los siguientes dispositivos móviles:

- Motorola Moto E 4G
  - CPU: Qualcomm Snapdragon 410 Quad-core 1.2 GHz Cortex-A53.
  - GPU: Adreno 306.

  - Chip Bluetooth 4.0 LE.
  - Sistema operativo: Android 5.0.2.
- Samsung Galaxy Alpha
  - CPU: 4 núcleos ARM Cortex-A15 a 1,8 GHz más 4 núcleos ARM Cortex-A7 a 1,3 GHz.
  - GPU: ARM Mali-T628 MP6.
  - Chip Bluetooth 4.0 LE.
  - Sistema operativo: Android 5.0.2.

5.4.2 SOFTWARE UTILIZADO PARA EL DESARROLLO

Para llevar a cabo el desarrollo de la aplicación, se ha hecho uso de una serie de herramientas software. A continuación, se listará cada una de ellas:

- ✓ Como entorno de desarrollo o IDE, se ha utilizado la herramienta oficial que propone Google y que ellos mismos han desarrollado. Esta herramienta es Android Studio. A fecha de redacción de este documento, la versión de la que se dispone es la 2.2.3. El link de acceso a la descarga es el siguiente: https://developer.android.com/studio/index.html?hl=es-419
- ✓ El SDK del que haremos uso también será el que propone Google para desarrollar en la plataforma Android, aunque no será necesario descargarlo ya que viene incluido en Android Studio.
- ✓ Para desarrollar el proyecto se ha hecho uso del lenguaje de programación Java, por tanto, necesitamos el JDK (Java Development Kit) que nos ofrece Oracle, que es la propietaria de este kit de desarrollo. A fecha de creación de este documento, va por la versión 8. El link de acceso a la descarga es el siguiente: http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

5.4.3 ESTRUCTURA DE NUESTRO PROYECTO ANDROID

La estructura de nuestro proyecto Android está compuesta por una serie de ficheros y directorios de tal forma que los elementos y el código queden bien

estructurados. Por una parte, tenemos la estructura que tienen todas las aplicaciones que se desarrollan en Android Studio, pero dentro de ella, el contenido es diferente (paquetes, clases, recursos, etc.). A continuación, se ofrece la primera vista de la estructura de un proyecto Android:

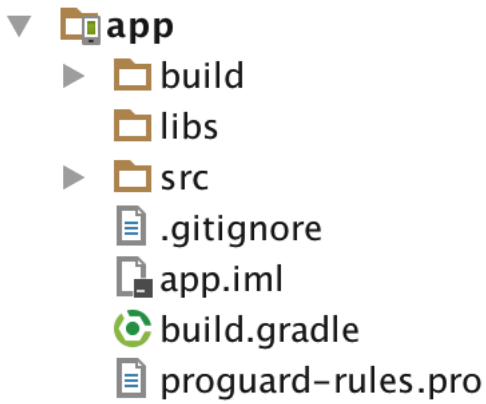

ILUSTRACIÓN 28: ESTRUCTURA PROYECTO ANDROID

A continuación, se ofrece una tabla con la explicación de los archivos y directorios mostrados anteriormente.

TABLA 26: DESCRIPCIÓN ESTRUCTURA PROYECTO ANDROID

App	**Módulo de nuestro proyecto.**
/build	Contiene resultados de compilación.
/libs	Contiene bibliotecas privadas.
/src	Contiene todos los archivos de código y recursos para el módulo.
.gitignore	Archivo de configuración de GIT.
App.iml	Almacena información sobre el módulo de desarrollo.
build.gradle	Define las configuraciones de compilación específicas para el módulo.
proguard-rules.pro	Archivo de configuración para utilizar proguard en nuestro proyecto.

A continuación, pasaremos a ver el contenido del directorio **/src** que es donde se almacena la mayor información del proyecto:

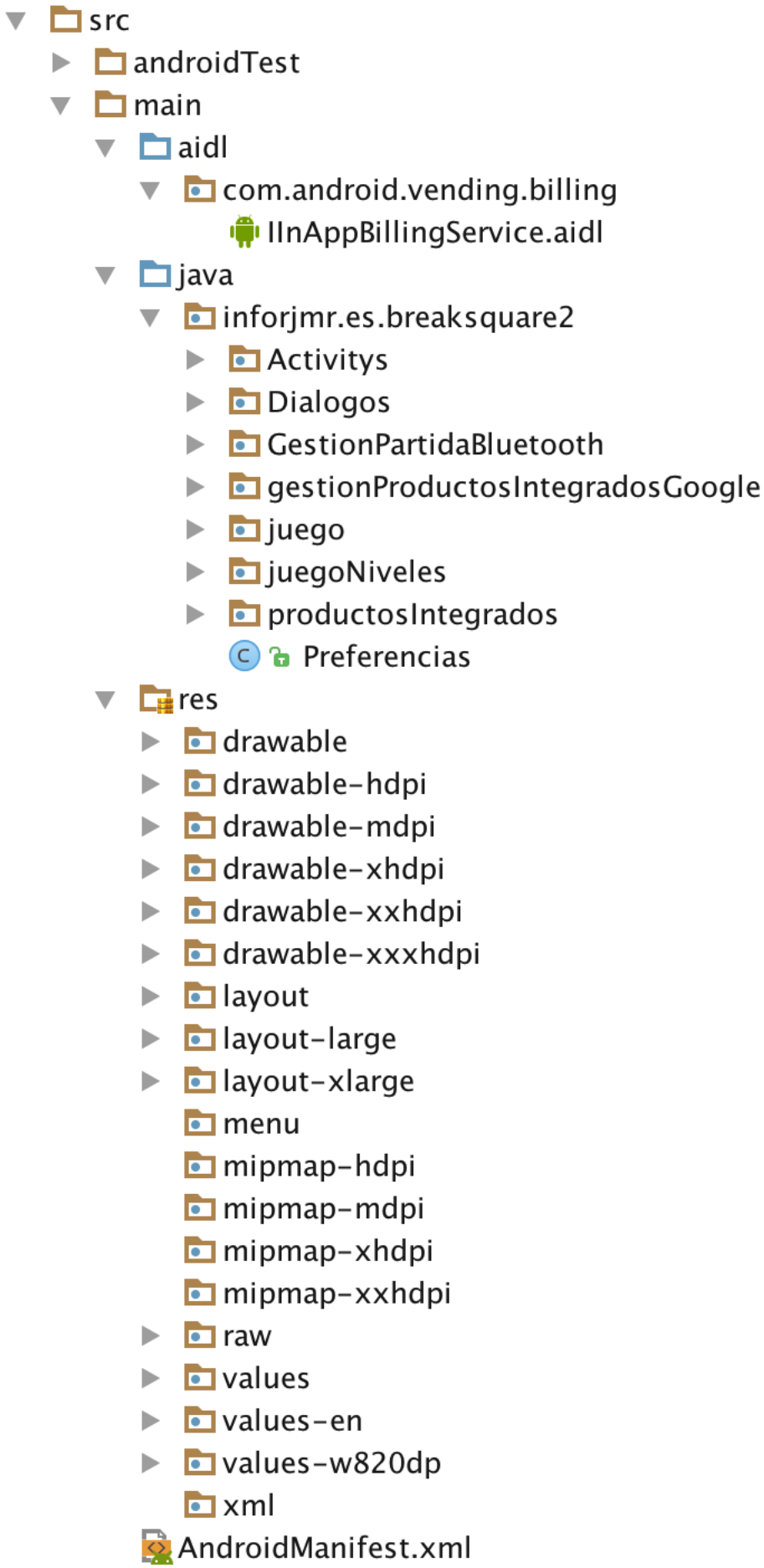
src
androidTest
main
aidl
com.android.vending.billing
IInAppBillingService.aidl
java
inforjmr.es.breaksquare2
Activitys
Dialogos
GestionPartidaBluetooth
gestionProductosIntegradosGoogle
juego
juegoNiveles
productosIntegrados
Preferencias
res
drawable
drawable-hdpi
drawable-mdpi
drawable-xhdpi
drawable-xxhdpi
drawable-xxxhdpi
layout
layout-large
layout-xlarge
menu
mipmap-hdpi
mipmap-mdpi
mipmap-xhdpi
mipmap-xxhdpi
raw
values
values-en
values-w820dp
xml
AndroidManifest.xml

ILUSTRACIÓN 29: CONTENIDO DEL DIRECTORIO SRC

Como se puede observar en la imagen anterior, el primer directorio es "**androidTest**", el cual contiene código para realizar pruebas en nuestra aplicación. A continuación, está el directorio "**main**" que por una parte contiene el directorio "**aidl**" que contiene la librería para integrar las compras en la aplicación, por otra parte, contiene el directorio "**java**" en el cual estará todo el código Java de nuestra aplicación, a continuación, tenemos el directorio "**res**" que contiene recursos que utiliza nuestra aplicación, tales como imágenes y estilos, layouts, cadenas de texto en uno o varios idiomas, sonidos, etc. Y por último se puede observar el archivo "**AndroidManifest.xml**" que proporciona información esencial sobre nuestra aplicación Android, que el sistema tendrá en cuenta para poder ejecutar el código de la APP.

A continuación, se muestra una tabla de cada paquete que se encuentra en el directorio java en la que explicaré qué es lo que contiene:

TABLA 27: DIRECTORIOS ESPECÍFICOS DE NUESTRA APLICACIÓN

activitys	Contiene todas las activitys por las que estará formado el juego.
dialogos	Contiene todos los diálogos que aparecerán en los diferentes escenarios del juego.
gestionPartidaBluetooth	Contiene todas las clases necesarias en la comunicación Bluetooth para el modo de juego en equipo.
gestiónProductosIntegradosGoogle	Contiene clases de apoyo que ofrece Google para realizar la compra de productos integrados.
juego	Contiene todas las clases necesarias de las que consta el juego.

juegoNiveles	Contiene todas las clases de todos los niveles que tiene el juego.
productosIntegrados	Contiene las clases para llevar a cabo la copra de los productos integrados de la aplicación.
preferencias.java (archivo)	Contiene todas las preferencias del juego.

A continuación, se muestra una tabla con el significado de los directorios que están dentro del directorio de recursos "**res**":

TABLA 28: CONTENIDO DEL DIRECTORIO "RES"

Directorio.	***Tipo de recurso.***
drawable/	Contiene archivos de mapas de bit (.png, .9.png, .jpg, .gif) o archivos XML que se han compilado en otros subtipos de recursos de elemento de diseño como formas, listas de estado, etc.
layout/	Archivos XML que definen el diseño de una interfaz de usuario.
menu/	Archivos XML que definen menús de aplicaciones, como un menú de opciones, un menú contextual o un submenú.
raw/	Archivos arbitrarios para guardar sin procesar.
values/	Archivos XML que contienen valores simples, como strings, valor enteros y colores.
xml/	Archivos XML arbitrarios que se pueden leer en tiempo de ejecución

5.5 FICHEROS FUENTES RELEVANTES DEL PROYECTO.

En este apartado se explican de forma más detallada algunas de las clases y métodos que, considero más importantes del proyecto. Aunque siempre se ha intentado que el código sea lo más limpio y simple posible para que se

entienda fácilmente. También se hará uso de la documentación interna, para facilitar su entendimiento.

5.5.1 CLASE GESTIONPRODUCTOINTEGRADO

Esta clase es instanciada y llamada a sus métodos cuando se quiere comprar un producto integrado en la aplicación. El producto integrado del que dispone el juego es para eliminar la publicidad que hay en él.

Dicha clase implementará las dos interfaces que se ven a continuación:

```
public class GestionProductoIntegrado implements
IabHelper.OnIabSetupFinishedListener,
IabHelper.OnIabPurchaseFinishedListener{
```

Dichas interfaces facilitarán la gestión de la compra del producto integrado implementando sus métodos.

Primeramente, necesitaremos cargar el billingHelper como se muestra a continuación en el método al que llamaremos para iniciar la compra del producto integrado:

```
/**
 * Método para inicializar la compra del producto integrado de la
aplicación. En él se inicializa el billingHelper (de tipo IabHelper) para
facilitar el proceso. En la última linea se inicia el proceso de compra,
cargando primeramente el bilingHelper
 * @param context
 */
public void startBuyProcess(Context context){
    String clave = "MIIBIjANBgkqhkiG9w...";
    billingHelper = new IabHelper(context, clave);
    billingHelper.startSetup(this);
}
```

Lo más llamativo de este método es la clave. Dicha clave, es facilitada por Google y pertenece a la clave de licencia de la aplicación en Google Play y que nos servirá también para realizar compras de productos integrados. Para encontrarla hay que ir a **Google Play Developer Console > Seleccionamos la apliación > apartado "Servicios y APIs.**

Una vez que se ha cargado el proceso anterior, cuando finaliza, llama automáticamente al siguiente método:

```
/**
 * Método llamado automáticamente cuando termina de cargarse billingHelper.
En él, se comprueba si ese elemento ha sido comprado con anterioridad (en
caso de que sólo haga falta comprarlo una vez) y en caso de que haya sido
comprado, elimina la publicidad y en caso contrario, inicia la compra del
producto integrado.
 * @param result The result of the setup process.
 */
@Override
public void onIabSetupFinished(IabResult result) {
    if (result.isSuccess()) {
        try{
            if(billingHelper.queryInventory(true, null).hasPurchase("1")){
                Toast.makeText(context, "¡El componente ya ha sido
comprado...  eliminando publicidad!", Toast.LENGTH_SHORT).show();
                quitarPublicidad();
            } else {
                compraElemento();
            }

        } catch(IabException e){
            e.printStackTrace();
        }

    } else {

        errorAlIniciar();
    }
}
```

Hay productos integrados que se compran una vez, como la eliminación de la publicidad en el que el límite de compras es 1 por cuenta de usuario o que pueden ser comprados varias veces, por ejemplo, en un juego que se compran monedas y no hay límites de compra. En este caso el límite de compras es 1 ya que se paga una vez y se elimina la publicidad. Sería como adquirir una versión “pro” del juego. Sería injusto que un usuario pagase por la versión sin publicidad del juego, lo desinstalase, y cuando lo volviese a instalar tuviera que volver a pagar por eliminar la publicidad. Eso es lo que se evita en este método, en primer lugar, comprobaría si el elemento ha sido comprado con anterioridad y en el caso de ser así, eliminaría la publicidad. En caso contrario iniciaría la compra del mismo.

Para comprobar si ha sido comprado con anterioridad, se hace a través de la siguiente consulta:

```
if(billingHelper.queryInventory(true, null).hasPurchase("1"))
```

En la consulta anterior, lo que más llama la atención es el número “1” este número, corresponde al número de identificación del producto integrado que

se quiere comprobar si fue comprado. Dicho número se asigna (lo asigna el desarrollador) al crear un nuevo producto integrado en Google Play Developer Console como aparece en la siguiente ilustración:

ILUSTRACIÓN 30: PRODUCTO INTEGRADO PARA UNA APLICACIÓN

Como hemos dicho anteriormente, si este producto no ha sido comprado con anterioridad, se inicia el proceso de compra el cual lo gestiona el siguiente método:

```
/**
 * Método que inicia el proceso de compra del producto integrado de nuestra
aplicación.
 */
protected void compraElemento() {
    billingHelper.launchPurchaseFlow(activity,"1",123,this);
}
```

Como se observa en el código anterior, el encargado de hacer esta gestión vuelve a ser billingHelper a través del método launchPurchaseFlow. Los parámetros que utiliza dicho método son; la actividad desde la que se va a comprar el elemento, el id del producto integrado, un código de respuesta (que asigna el desarrollador) que nos será de utilidad en caso de tener varios productos integrados y el último parámetro será el contexto.

Se inicia el proceso de compra y al usuario le aparece una ventana dónde se le pregunta si quiere realizar la compra y al finalizar esta interacción con el usuario, se llama al método siguiente:

```
/**
 * Gestiona el resultado del proceso de compra al finalizar este mismo.
 * @param result The result of the purchase.
 * @param info The purchase information (null if purchase failed)
 */
@Override
```

```
public void onIabPurchaseFinished(IabResult result, Purchase info) {
    if (result.isFailure()) {
        Toast.makeText(context, "Compra fallida, vuelva a intentarlo más
tarde.", Toast.LENGTH_SHORT).show();
    } else if (info.getSku().equals("1")) {
        Toast.makeText(context, "¡Gracias por colaborar, publicidad
eliminada!", Toast.LENGTH_SHORT).show();
        quitarPublicidad();
    }
}
```

En el código anterior se gestiona si el usuario ha realizado la compra correctamente o si ha habido un error por medio para poder informarle de que el producto no se ha comprado. Uno de los detalles más interesantes de este código es el siguiente:

```
if (info.getSku().equals("1"))
```

Dicho código es el que consulta si el producto integrado que se ha comprado, pertenece al id con el que lo estamos comparando y poder actuar en consecuencia eliminando la publicidad.

Por último, después de comprar, el producto integrado, se llama al método para eliminar la publicidad, que es el siguiente:

```
/**
 * El siguiente método se encarga de hacer que no aparezca más la
publicidad. Para ello, en el fichero de preferencias asigna "verPublicidad =
false"
 */
private void quitarPublicidad(){
    SharedPreferences prefs =
context.getSharedPreferences("preferenciaspuntuaciones",
Context.MODE_PRIVATE);
    SharedPreferences.Editor editor = prefs.edit();
    editor.putBoolean("verPublicidad", false);
    editor.commit();
}
```

5.5.2 CLASE JUEGOACTIVITY

Esta clase, como su nombre indica, gestionará la activity del Juego, es decir la interfaz gráfica de todo lo que pasa en el juego, desde mostrar el estado del juego en cada momento, hasta mostrar la publicidad, así como la gestión que hará el usuario de la partida Bluetooth y los diálogos que aparecerán en dicho juego. Por ello, es una de las clases más amplias e importantes del juego.

A continuación, se explicarán algunos de los métodos que más llaman la atención en dicha clase.

5.5.2.1 MÉTODO BLUETOOTHACTIVADO

```
/**
 * El siguiente método comprueba si un dispositivo contiene el requisito de
ser compatible con Bluetooth BLE y en caso de ser compatible, hacer las
gestiones necesarias para iniciarlo si no está iniciado.
 *
 */
private void bluetoothActivado(){
    bluetoothActivado = false;
    if
(!getPackageManager().hasSystemFeature(PackageManager.FEATURE_BLUETOOTH_LE))
{
        Toast.makeText(this, "Este dispositivo no soporta BLE, modo de juego
por equipos no disponible.", Toast.LENGTH_LONG).show();
        finish();
    }else{

        // Initializes Bluetooth adapter.
        final BluetoothManager bluetoothManager = (BluetoothManager)
getSystemService(Context.BLUETOOTH_SERVICE);
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.JELLY_BEAN_MR2) {
            mBluetoothAdapter = bluetoothManager.getAdapter();

        }

        if (mBluetoothAdapter == null || !mBluetoothAdapter.isEnabled()) {
            Intent enableBtIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
            startActivityForResult(enableBtIntent, REQUEST_ENABLE_BT);
        }else{
            bluetoothActivado = true;
        }

    }

}
```

El método anterior sólo se ejecuta cuando el usuario selecciona el modo de juego en equipo, ya que es el único modo de juego que necesitará de la utilización del Bluetooth para poder llevar a cabo la partida. Por una parte, comprueba que el dispositivo cumpla el requisito de tener Bluetooth LE y una vez que se cumple dicho requisito, se inicia el proceso de comprobación del Bluetooth para ver si está activado o no. En caso de no estar activado se procede con la solicitud de la activación a través del siguiente código:

```
Intent enableBtIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
startActivityForResult(enableBtIntent, REQUEST_ENABLE_BT);
```

El código anterior, inicia el proceso para la activación del Bluetooth por parte del usuario.

5.5.2.2 MÉTODO MOSTRARANUNCIO

```
/**
 * Método que gestiona que se muestre un anuncio. El anuncio se mostrará
cada 3 partidas jugadas y dicha cuenta se gestionará en este método con
ayuda de las preferencias, en la que se irá almacenando la cuenta.
 * @return devuelve true, en caso de haber mostrado el anuncio y false en el
caso contrario.
 */
public boolean mostrarAnuncio(){
    boolean mostrarAnuncio = false;
    int recuentoPublicidad = preferencias.getCuentaAtrasPublicidad();

    if(recuentoPublicidad == 0){
        preferencias.setCuentaAtrasPublicidad(3);
        runOnUiThread(new Runnable() {
            @Override
            public void run() {
                if(interstitialAd.isLoaded() &&
preferencias.verPublicidad()) {
                    interstitialAd.show();
                }
            }
        });
        mostrarAnuncio = true;
    }else{
        preferencias.setCuentaAtrasPublicidad(recuentoPublicidad - 1);
    }
    return mostrarAnuncio;
}
```

En el código anterior se gestiona el mostrar un anuncio obteniendo de las preferencias el número de partidas jugadas y ver si se debe mostrar dicho anuncio. Si se debe mostrar el anuncio, primero se hace una comprobación de si el anuncio ha sido cargado (iterstitialAd.isloaded) y si es así, lo muestra. El anuncio se carga en el onCreate de la actividad de la siguiente forma:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.layout_juego);
...
    // Crear el intersticial.
    interstitialAd = new InterstitialAd(this);
    interstitialAd.setAdUnitId("ca-app-pub-0599487307354918/9911051253");
   // Crear la solicitud de anuncio.
    adRequest = new AdRequest.Builder().build();
    cargarAnuncio();

...
}

/**
 * Método encargado de cargar el anuncio solicitado
 */
```

```
public void cargarAnuncio(){
    new Thread(new Runnable() {
        @Override
        public void run() {
            runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    interstitialAd.loadAd(adRequest);
                }
            });

        }
    }).start();
}
```

Del código anterior hay que destacar que el interstitialAd necesita un id. Dicho id pertenece al anuncio en cuestión que el usuario tiene dado de alta en AdMob. Por tanto, para implementar anuncios en nuestra aplicación, debemos tener una cuenta de AdMob dada de alta. Dicha cuenta, se tiene que enlazar con las aplicaciones que tenemos subidas en Google Play que van a mostrar productos. A continuación, en la ilustración se muestra la aplicación con el bloque de anuncios y su id:

ILUSTRACIÓN 31: BLOQUE DE ANUNCIOS EN ADMOB

5.5.2.3 MÉTODO GUARDARPUNTUACIONPREFERENCIAS

En este método se gestiona el guardado de la puntuación que hace un usuario para que, al finalizar la partida, se pueda comparar la puntuación mayor almacenada con la obtenida en la partida e informar de si se ha batido un nuevo record.

El código siguiente muestra como guarda la puntuación obtenida en la partida siempre y cuando sea mayor que la almacenada:

```
/**
 * Método que guarda la puntuación obtenida en preferencias sólo, si es
mayor que la que estaba guardada.
 * @return Devuelve true si se ha batido el record, false si no se ha batido
 */
public boolean guardarPuntuacionPreferencias(int puntuacion){
    int mejorPuntuacion = preferencias.getMejorPuntuacion(); //Obtiene el
```

```
campo mejorPuntuacion que contiene la mejor puntuación hasta ahora

    if(mejorPuntuacion < puntuacion) { //Guarda la nueva puntuación si es
mejor que la anterior
        SharedPreferences prefs =
this.getSharedPreferences("preferenciaspuntuaciones", Context.MODE_PRIVATE);
        SharedPreferences.Editor editor = prefs.edit();
        editor.putInt("mejorPuntuacion", puntuacion);
        editor.commit();
        return true;
    }
    return false;

}
```

5.5.2.4 MÉTODO CALCULARALTOYLARGOPANTALLA

El siguiente método también es un método muy importante y sencillo a la vez. Dicho método se utiliza para calcular el alto y ancho de la pantalla del dispositivo para que una vez se proceda a inicializar el juego, este se adapte en proporción a lo ancho y alto del dispositivo en cuestión. El código es el siguiente:

```
/**
 * Método que guarda en las variables pertinentes, el alto y largo de la
pantalla del dispositivo en el que se está jugando.
 */
private void calcularAnchoLargoPantalla(){
    DisplayMetrics metrics = new DisplayMetrics();
    this.getWindowManager().getDefaultDisplay().getMetrics(metrics);
    largoPantalla = metrics.heightPixels;
    anchoPantalla = metrics.widthPixels;
}
```

Para obtener los tamaños, sólo necesitamos instanciar de la clase DisplayMetrics, obtener las métricas y a través de sus métodos, obtener el alto y el ancho de la pantalla.

5.5.2.4 MÉTODO LANZARDIALOGOFINDEJUEGOPUNTUACION

Del siguiente método, nos vamos a centrar en la parte en que se envía la puntuación al ranking de Google para que lo gestione. Lo demás nos lo vamos a saltar ya que es algo trivial, de fácil entendimiento y sin demasiada importancia.

```
/**
 * Método que envia la puntuación obtenida al Rankin y que muestra en forma
de diálogo dicha puntuación y si ha habido record.
 * @param puntuacion
```

```
 */
public void lanzarDialogoFinJuegoPuntuacion(int puntuacion){
...
    if(mGoogleApiClient.isConnected()) {
        Games.Leaderboards.submitScore(mGoogleApiClient,
getString(R.string.leaderboard_best_players), puntuacion);

    }
...
}
```

Como se puede observar en el código, previamente al envio de la puntuación, se hace una comprobación de si estamos conectados a la API de Google Play Games, dicha conexión se hace en el método onCreate de la actividad. El siguiente código pertenece a la conexión:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.layout_juego);
...
    // Create the Google Api Client with access to the Play Games services
    mGoogleApiClient = new GoogleApiClient.Builder(this)
            .addConnectionCallbacks(this)
            .addOnConnectionFailedListener(this)
            .addApi(Games.API)
            .addScope(Games.SCOPE_GAMES)
            .build();
    //FIN CONEXION GOOGLE GAMES
...
}
```

Volviendo al método y al envio de la puntuación, cabe destacar un dato importante y es que a la hora de hacer el envio, se necesita enviar un identificador, en el método no se aprecia porque dicho identificador está contenido en un string de un archivo xml al que se hace referencia (este procedimiento es una buena práctica para que el código sea más limpio), como se puede ver en la siguiente línea:

```
Games.Leaderboards.submitScore(mGoogleApiClient,
getString(R.string.leaderboard_best_players), puntuacion);
```

En el método submitScore, se necesitan 3 parámetros, por una parte, está la conexión con la API de Google Play Games, por otra parte, lo que se ha comentado anteriormente del identificador, y por último la puntuación obtenida.

Ahora pasamos a explicar de dónde obtenemos el identificador del que hemos estado hablando anteriormente, dicho identificador, pertenece al leaderboard dado de alta en la Google Play Developer Console, que se encuentra en el apartado **"Servicios de Juegos > App de la que queramos ver los marcadores > marcadores"**. A continuación, se muestra una imagen de los dos marcadores creados para nuestro juego, por una parte, el marcador para el modo de juego por nieves, y por otra, el marcado para el modo de juego clásico:

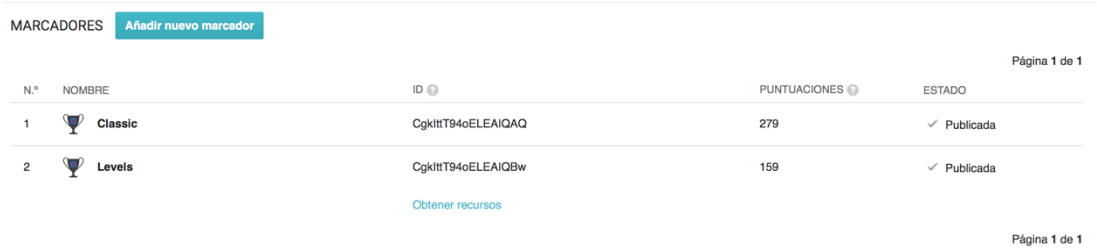

ILUSTRACIÓN 32: MARCADORES CREADOS PARA EL JUEGO

Como se puede observar en la ilustración anterior, están disponibles los identificadores necesarios para enviar la puntuación y dependiendo del identificador que pase como parámetro, la puntuación pertenecerá a un marcador u otro.

5.5.3 JUEGO

En este apartado, no vamos a hablar de una clase en concreto, se va a hablar del juego en conjunto (no de la app sino de la estructura del juego) ya que, al haber 3 modelos de juegos y 38 niveles, el diseño de la estructura es un detalle que se ha tenido muy en cuenta a la hora de programarlo para hacerlo lo más sencillo y reutilizable posible.

Por una parte, tenemos la clase "Juego", que es una clase abstracta que contiene todos los métodos que serán necesario en cualquier modo de juego o nivel. Por otra parte, están las clases "JuegoClasico", "JuegoNiveles" y "JuegoEquipo", que extenderán de la clase juego para poder reutilizar sus métodos y sobreescribirlos según sea necesario.

Por otra parte, tenemos las clases “Niveles2x2”, “Niveles3x3”, “Niveles5x5”, “Niveles7x7” que extienden de la clase JuegoNiveles.

Y por último tenemos las clases de todos los niveles desde la clase “Nivel1” hasta la clase “Nivel38” que extenderán de las anteriores según proceda. Para hacer más sencillo entender la herecia que se ha llevado a cabo, se ha realizado una representación en forma de árbol de la misma.

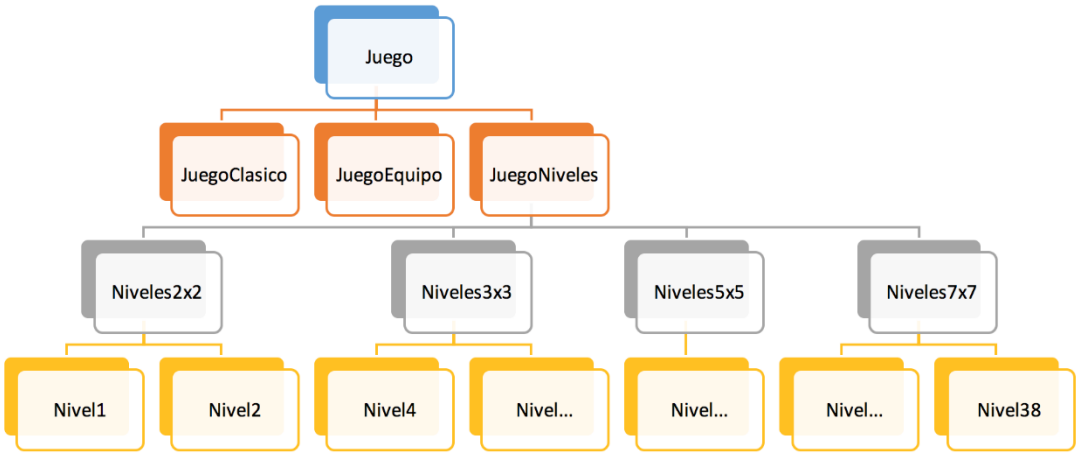

ILUSTRACIÓN 33: ESTRUCTURA DE LA HERENCIA DEL JUEGO

A continuación, se hablarán de los métodos que considero más importantes en las clases anteriores:

5.5.3.1 MÉTODO CALCULARCOORDENADASMINISQUARES

Como bien dice el nombre del apartado, el siguiente método lo que hace es asignar las coordenadas en las que aparecerán los cuadros pequeños de la parta superior de la pantalla, este método lo considero importante debido a que para cada dispositivo móvil las coordenadas calculadas serán diferentes dependiendo de algunos parámetros como los píxeles, densidad de la pantalla, número de filas y columnas, tamaño del MiniSquare etc.

A continuación, se muestra la codificación del método en cuestión:

```
/**
 * Método que calcula las coordenadas de la pantalla en las que aparecerán
los cuadros rojos.
 */
private void calcularCoordenadasMiniSquares(){

    //PosX del primer cuadro rojo
    int posX = (((anchoPantalla-(borde * (numFilasYColumnas -1)) -
```

```
(tamanioBigSquareAdaptado * numFilasYColumnas))/2) +
tamanioBigSquareAdaptado/2 - tamanioMiniSquareAdaptado /2);
    //PosY de todos los miniSquares rojos (30px)
    int posY = 30;

    for(int i=0; i<numFilasYColumnas; i++) {
        coordenadasMiniSquares[i] = new Coordenada(posX,posY);
        posX = posX+(tamanioBigSquareAdaptado + borde);
    }
}
```

Al igual que existe un método para los MiniSquares también existe otro para los BigSquares (cuadrados que estarán en la parte de debajo de la pantalla) el cual es muy parecido.

5.5.3.2 MÉTODO CONTROLCOLISION

Este método en la clase “Juego” es un método abstracto que todas las subclases tienen que implementar, he decidido crearlo como abstracto ya que en cada clase que lo implemente, deberá realizar una función un otra dependiendo del modo de juego. A continuación, muestro el código del control colisión del modo de juego clásico:

```
/**
 * Comprueba si un minisquare ha quedado dentro de un bigsquare para
eliminarlo o finalizar la partida.
 */
@Override
public void controlColision() {
    MiniSquare cuadro = this.getMiniSquares().get(0);
    int posActual = cuadro.getColumna();
    boolean encontrado = false;
    cuadro.parar();

    for(int i=0; i < 10; i++){
        if(bigSquares[posActual].getY() <= cuadro.getY() &&
(cuadro.getY()+cuadro.getTamanioX()) <= (bigSquares[posActual].getY() +
bigSquares[posActual].getTamanioX()) &&
bigSquares[posActual].getEstaActivada()){

            puntuacion = puntuacion + bigSquares[posActual].getPuntuacion();
            bigSquares[posActual].desactivarCelda();
            reactivarCeldas();
            encontrado = true;
        }
        posActual=posActual+10;
    }
    if(encontrado){
        miniSquares.remove(0);
        miniSquares.get(0).iniciar();
    }else{
        setFinalizarJuego(true);
    }
}
```

Lo que hace el código anterior, no es más que lo que pone en su descripción en la documentación interna; comprueba si un minisquare ha quedado dentro de un bigsquare y actúa en consecuencia dependiendo del resultado, bien eliminando el bigsquare y el minisquare en caso de que este último haya quedado dentro (encontrado=true) o finaliza la partida.

El funcionamiento del método es el siguiente; Se sabe la columna del minisquare, así que dependiendo de su posición y de la posición de los bigsquare dentro de esa columna, va comprobando uno a uno si la coordenada cae dentro de alguno de ellos, si es así, se da por eliminado el bigsquare, y sigue la partida iniciándose para que caiga un nuevo minisquare, en caso contrario, finaliza.

5.5.3.3 MÉTODO REDISTRIBUIRCELDA

Este método sólo está en los niveles, y su función es redistribuir las celdas según se deseen poner dependiendo del nivel en el que se encuentre.

El código siguiente muestra el código del nivel 3:

```
/**
 * Redistribuye las bigsquares a dos azules ( los dos superiores ) y dos
rojos (los dos inferiores)
 */
@Override
void redistribuirCeldas() {
    BigSquare bigSquares[] =  this.getBigSquares();
    bigSquares[0].desactivarCeldaARoja();
    bigSquares[1].desactivarCeldaARoja();

    ArrayList<MiniSquare> miniSquares;
    miniSquares = this.getMiniSquares();

    miniSquares.remove(1);
    miniSquares.remove(0);

}
```

5.5.4 CLASE GESTIONBEACON

Esta clase, sólo instanciada en el modo de juego en equipo, es la encargada de gestionar tanto los beacon entrante como los salientes. Por ella pasan

todos los beacons que gestiona nuestro dispositivo, y los procesa según el contenido de sus mensajes.

5.5.4.1 MÉTODO PROCESARBEACONENTRADA

El método procesarBeaconEntrada es uno de los métodos más importantes de esta clase ya que por este método pasan todos los beacon que se detectan, pertenezcan a la partida o no. A continuación se muestra el código del método:

```
/**
 * Procesa todos los beacons que detecta el dispositivo y  los procesa según
pertenezcan a la partida o no y según el tipo de mensaje que contengan.
 * @param b
 */
public void procesarBeaconEntrada(Beacon b) {

    if (!beaconsProcesados.contains(b)) {
        if (getDirBluetooth1(b).equals(direccionBluetoothMaestro) ||
getDirBluetooth1(b).equals(miDireccionBluetooth) || b.getId2().toInt() ==
BEACON_NUEVA_PARTIDA) {
            if (idPartida.equals((b.getId3() + "")) || b.getId2().toInt() ==
BEACON_NUEVA_PARTIDA) {
                switch (b.getId2().toInt()) {
                    case BEACON_NUEVA_PARTIDA:
                        if (!esMaestro) {
                            procesarBeaconNuevaPartida(b, partidas);
                        }
                        beaconsProcesados.add(b);
                        break;
                    case BEACON_TU_TURNO:
                        if
(miDireccionBluetooth.equals(getDirBluetooth1(b))) {
                            gestionJuegoBeacons.activarTurno();
                            gestionJuegoBeacons.lanzarMiniSquare();
                        }
                        beaconsProcesados.add(b);
                        break;
                    case BEACON_ULTIMO_TURNO:
                        if
(direccionBluetoothMaestro.equals(getDirBluetooth1(b))) {

gestionJuegoBeacons.desactivarBigSquare(getBigSquareEliminado(b));
                        }
                        beaconsProcesados.add(b);
                        break;
                    case BEACON_FIN_PARTIDA:
                        if
(direccionBluetoothMaestro.equals(getDirBluetooth1(b))) {
                            gestionJuegoBeacons.finalizarJuego();
                        }
                        beaconsProcesados.add(b);
                        break;
                    case BEACON_FIN_TURNO:
                        if (getBigSquareEliminado(b) != -1) {
                            lanzarBeaconUltimoTurno(getDirBluetooth2(b),
getBigSquareEliminado(b));

gestionJuegoBeacons.desactivarBigSquare(getBigSquareEliminado(b));
                            nuevoTurno();
                        } else {
                            gestionJuegoBeacons.lanzarBeaconFinPartida();
                            gestionJuegoBeacons.finalizarJuego();
```

```
                        }
                        beaconsProcesados.add(b);
                        break;
                    case BEACON_UNIRSE_A_PARTIDA:
                        if
(miDireccionBluetooth.equals(getDirBluetooth1(b))) {
                            procesarUnirsePartida(b, jugadores);
                        }
                        beaconsProcesados.add(b);
                        break;
                }
            }
        }
        beaconsProcesados.add(b);
    }

}
```

Como se puede observar, en el código anterior, cuando llega un nuevo beacon, lo primero que se hace es comprobar si dicho beacon ya ha sido procesado para no volverlo a procesar. En segundo lugar, se comprueba si es un beacon anunciando una partida, si pertenece a la partida o si el que lo envía es el creador de la partida. Comparar si es el id de partida a la que pertenecemos y el creador de la partida a la que pertenecemos, se realiza porque podría darse el caso de que dos partidas diferentes tuvieran el mismo id y estuvieran en la misma zona, es difícil (posibilidad del 0.01%), pero podría darse el caso. Por último, procesaría el mensaje según el tipo de mensaje que fuese. Los tipos de mensajes se han definido de la siguiente forma:

```
private static final int BEACON_NUEVA_PARTIDA = 1;
private static final int BEACON_UNIRSE_A_PARTIDA = 2;
private static final int BEACON_ULTIMO_TURNO = 3;
private static final int BEACON_TU_TURNO = 4;
private static final int BEACON_FIN_TURNO = 5;
private static final int BEACON_FIN_PARTIDA = 6;
```

5.5.4.2 MÉTODO TRANSMITIRBEACON

La función de este método como su nombre indica es transmitir un beacon, para implementarlo se ha tenido en cuenta que el creador de la partida va a tener dos flujos de transmisión de beacons el primer flujo para mostrar quién cuál ha sido el último bigSquare eliminado en el último turno y el segundo flujo para la gestión de la partida (BEACON_TU_TURNO, BEACON_FIN_PARTIDA). A continuación, se muestra cuál es el código del método.

```
private  void transmitirBeacon(String id1, String id2){
    Log.i("id1",id1);
    Beacon beacon;
    if(id2.equals("1")){
        idPartida = generarIDpartida();
    }
    beacon = new Beacon.Builder()
            .setId1(id1)
            .setId2(id2)
            .setId3(idPartida)
            .setManufacturer(0x0118)
            .setTxPower(-56)
            .setDataFields(Arrays.asList(new Long[] {0l}))
            .build();

    if(Integer.parseInt(id2)==3){//Si es un beacon de útlimo turno

        if(transmitirBeaconUltimoTurno != null){

            transmitirBeaconUltimoTurno.pararTransimisionBeacon();
        }
        transmitirBeaconUltimoTurno= new TransmitirBeacon();
        transmitirBeaconUltimoTurno.execute(beacon);


    }
    else{
        if(transmitirBeaconGeneral != null){

            transmitirBeaconGeneral.pararTransimisionBeacon();
        }
        transmitirBeaconGeneral= new TransmitirBeacon();
        transmitirBeaconGeneral.execute(beacon);
    }

}
```

Como podemos ver en el código anterior, el método es sencillo y fácil de entender, pero como se puede comprobar, para transmitir (literalmente) el beacon, se hace uso de la instancia de una clase privada que hemos creado. El código de la clase privada es el siguiente:

```
private class TransmitirBeacon extends AsyncTask<Beacon, Beacon, Void>{

    Beacon beacon;
    BeaconTransmitter beaconTransmitter;
    @Override
    protected Void doInBackground(Beacon... params) {
        beacon = params[0];
        beaconTransmitter = new BeaconTransmitter(context,beaconParser);

        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP) {
            beaconTransmitter.startAdvertising(beacon, new
AdvertiseCallback() {

                @Override
                public void onStartFailure(int errorCode) {
                    Log.e(TAG,"Advertisement start failed with code: " +
errorCode);
                }

                @Override
```

```
                public void onStartSuccess(AdvertiseSettings
settingsInEffect) {
                    Log.i(TAG, "Advertisement start succeeded.");
                }

            });
        }
        return null;
    }
```

La clase anterior extiende de un AsyncTask para ejecutar el lanzamiento del beacon en segundo plano. Para ello se crea una instancia de la clase **"BeaconTransmiter"** y se hace uso de su método **"startAdvertising()"**.

5.5.5 CLASE GESTIONJUEGOBEACON

La clase "GestionJuegoBeacon" es una clase que hace de puente entre el juego y la gestión de beacon. Dicha clase se utiliza para transformar los mensajes que se reciben del exterior de otros dispositivos en órdenes para el juego, por ejemplo, para informar de que se tiene el turno o de que la partida ha terminado. También sirve para enviar mensajes de acciones que suceden en el juego hacía otros dispositivos en forma de beacon, por ejemplo, cuando se elimina un bigSquare o cuando se termina el turno y se quiere informar al creador de la partida.

En dicha clase no hay ningún método que sea extenso o de difícil comprensión, ya que todos los métodos, de lo único que se encargan es de pasar información de una clase a otra, sin que se procese mucha información en ellos.

5.6 CONCLUSIONES

Como conclusión a este capítulo, decir que también se ha superado de forma exitosa, he nombrado y justificando en todo momento porqué he decidido tomar unas decisiones u otras, dependiendo de los conocimientos que he adquirido tanto en la carrera como a la hora de investigar los temas que he investigado. También he explicado y desarrollado las partes de la

programación que me han resultado de mayor interés, para facilitar la comprensión de la estructura interna del juego.

En el próximo capítulo, pasaremos a explicar cómo llevar a cabo la instalación del APK del juego y cómo utilizarlo.

6 MANUAL DE USUARIO

En este nuevo capítulo de la documentación se tratará todo lo relacionado con el manual de usuario. Dicho manual de usuario será de utilidad a toda aquella persona que instale y ejecute el juego por primera vez, facilitando la navegación por el mismo.

En primer lugar, se describirá en qué consiste el juego. A continuación, se nombrarán los requisitos que el dispositivo debe cumplir para que el juego sea completamente funcional. En tercer lugar, se explicará cómo instalar el juego. Por último, se mostrará cómo navegar a través del juego.

6.1 DESCRIPCIÓN DEL JUEGO

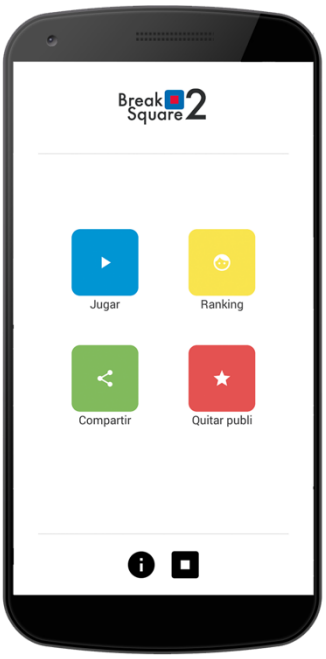

ILUSTRACIÓN 34: VISTA PRINCIPAL DE BREAK SQUARE

Break Square es un juego sencillo y adictivo para dispositivos (móviles y tabletas) Android, ideal para pasar un buen rato. La temática de Break Square 2 es sencilla. Hay dos tipos de cuadrados, los rojos y pequeños, y los grandes. Los cuadrados rojos y pequeños estarán colocados de forma horizontal en la parte superior de la pantalla y los cuadros grandes formarán una matriz cuadrada de X cuadrados que estarán situados en la parte inferior de la pantalla. Los cuadrados rojos irán cayendo uno a uno a velocidad y orden aleatorio y si pulsas sobre cualquier parte de la pantalla se pararán. El juego consiste en pulsar la pantalla y parar los cuadrados rojos pequeños dentro de los cuadrados grandes azules, para eliminar estos últimos.

Para entender mejor la mecánica del juego, puedes ver elvideo de presentación de una version anterior: https://youtu.be/6N4Q7k-AyHM

6.2 REQUISITOS DEL JUEGO

Para poder disfrutar del juego completo, el dispositivo que lo instale debe cumplir dos requisitos fundamentales, ya que, si no los cumple, el juego no sería jugable en alguno de sus modos de juego. Dichos requisitos son los siguientes:

1. Smartphone o Tablet con sistema operativo Android en su versión 5.0 o posterior.
2. Smartphone o Tablet con chip Bluetooth 4.0 Low Energy.

6.3 INSTALACIÓN

En este apartado explicaremos cómo instalar la aplicación. Lo primero que se tiene que hacer es almacenar el archivo instalable, APK, en el teléfono. A continuación, y con un gestor de archivos, buscamos el instalable de BreakSquare en el lugar donde lo hayamos guardado. Como gestor de archivos, en este manual se utiliza la aplicación "ES File Explorer", la podemos descargar en el siguiente enlace: https://play.google.com/store/apps/details?id=com.estrongs.android.pop&hl=es. En la siguiente ilustración, se muestra el instalable buscado con el explorador de archivos.

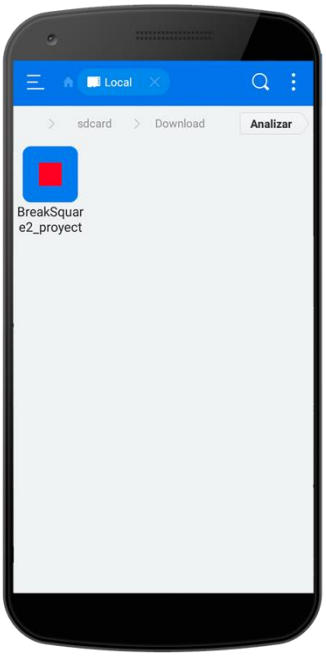

ILUSTRACIÓN 35: VISTA ES FILE EXPLORER

A continuación, se pulsará sobre el archivo, nos aparecerán sus propiedades y la opción de instalarlo. Pulsamos sobre el botón "Instalar" para instalarlo. Seguidamente nos aparecerá un diálogo de información. Dicho diálogo, nos informa de que no tenemos activada la opción para poder instalar aplicaciones que no hayan sido descargadas de Google Play, si dicha opción la hemos

activado antes, este cuadro no nos aparecerá. A continuación, se muestra una ilustración del proceso anterior:

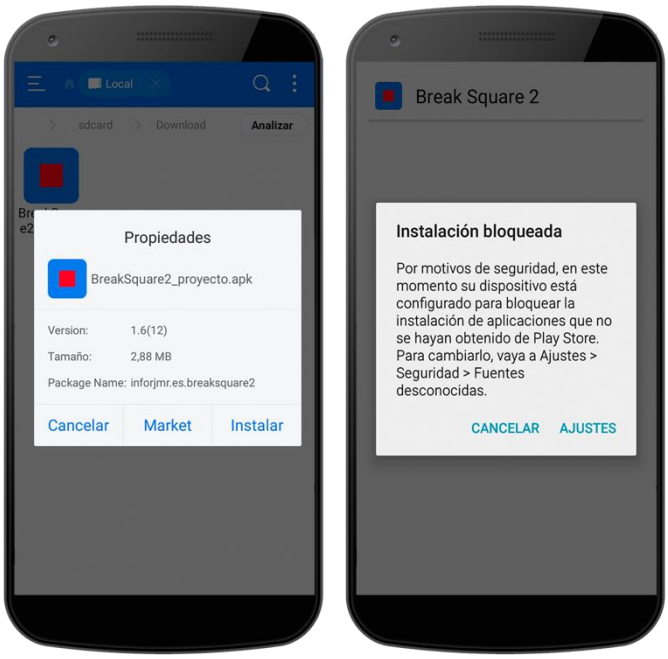

ILUSTRACIÓN 36: VISTA PROCESO DE INSTALACIÓN I

Al no tener la opción nombrada anteriormente activada, procederemos a activarla. Para ello pulsaremos en el botón “AJUSTES” y se nos abrirán los ajustes para poder activar la opción “Fuentes desconocidas”. Al activarla, se nos pregunta si sólo queremos activarla para la instalación actual, o para todas las instalaciones que se hagan en el dispositivo, seleccionar esta opción o no, depende del uso que le quiera dar el usuario a su dispositivo, pero por seguridad, se recomienda activarla sólo para esta instalación. A continuación, se muestran las imágenes del proceso:

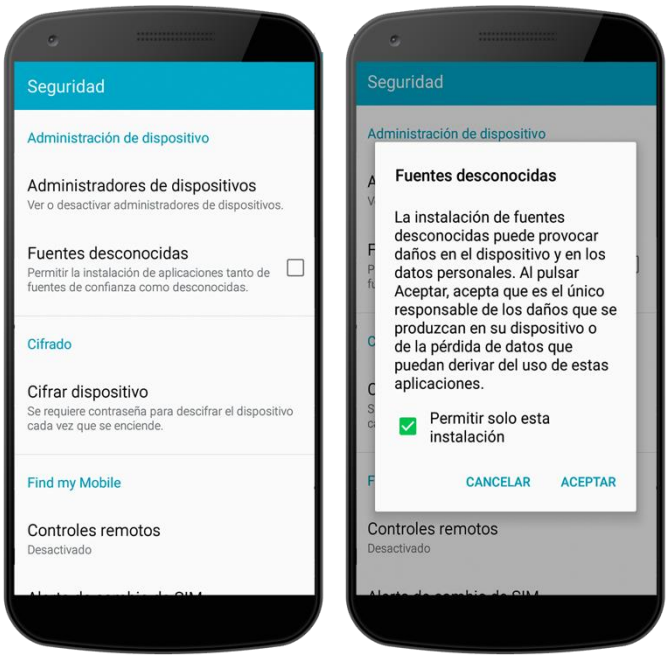

ILUSTRACIÓN 37: VISTA PROCESO DE INSTALACIÓN II

Una vez dado el permiso para poder instalar aplicaciones que no provengan de Google Play, se nos volverá a preguntar si queremos instalar la aplicación, informando de los accesos o permisos que tendrá la misma. Para instalarla pulsamos sobre el botón “INSTALAR” y comenzará a instalarse. Una vez instalada, nos dará la opción de abrirla. A continuación, se muestra la imagen del proceso:

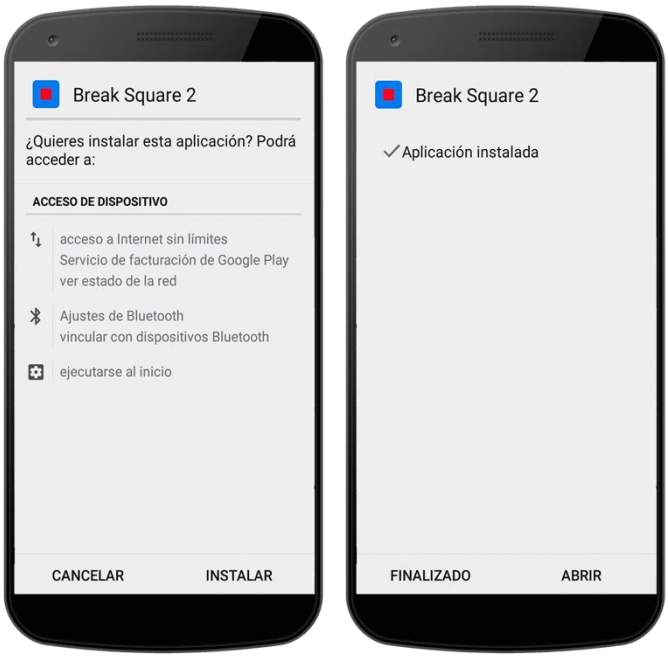

ILUSTRACIÓN 38: VISTA PROCESO DE INSTALACIÓN III

6.4 EJECUCIÓN DE LA APLICACIÓN

En este apartado, mostraremos como navegar a través del juego. La interfaz del juego es muy sencilla e intuitiva por lo que navegar por ella, será una tarea fácil.

Una vez que iniciamos la aplicación la primera vez, se nos solicitará conectar con nuestra cuenta de Google en Google Play Games, también aparecerá una vista de un diálogo informando sobre el juego y al cerrar ambas cosas, veremos la vista principal del juego. A continuación, se muestran dichas vistas:

A continuación, en la vista que se nos queda, tenemos varias posibilidades:

1. Pulsar sobre el botón “Jugar”. Nos llevará a ver los modos de juego y poder jugar a uno de ellos. A continuación, se muestra la vista resultante:

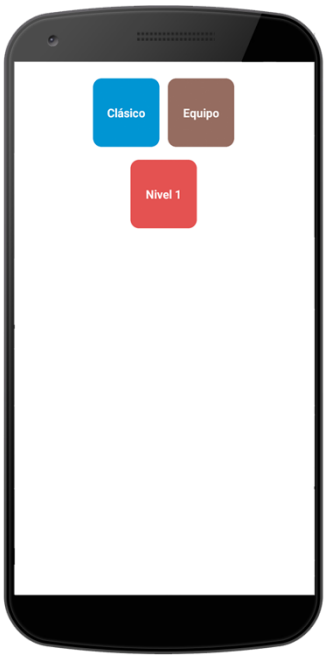

ILUSTRACIÓN 39: VISTA MODO DE JUEGOS BREAK SQUARE

En la vista anterior tenemos 3 posibilidades más:

2. Pulsar sobre el modo de juego “Clásico”. Se nos mostrará el inicio del modo de juego en modo clásico.

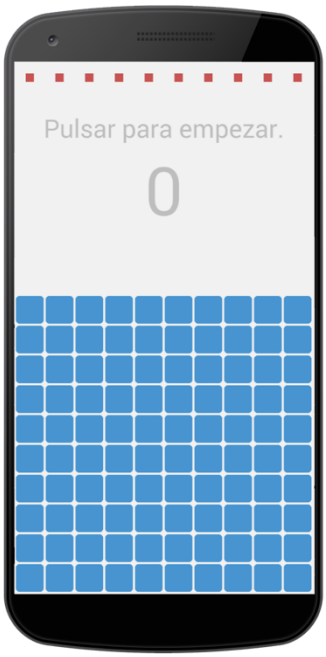

ILUSTRACIÓN 40: VISTA MODO DE JUEGO CLÁSICO I

Cuando se pulse para empezar, empezarán a caer los cuadrados superiores de uno en uno con velocidad y orden

aleatorio. La partida finalizará cuando uno falle. La partida irá avanzando de forma similar a la de la siguiente vista:

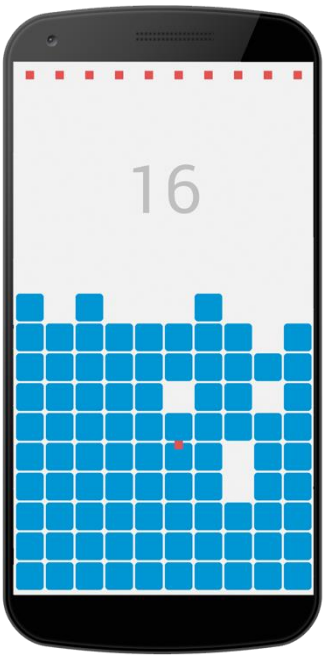

ILUSTRACIÓN 41: VISTA MODO DE JUEGO CLÁSICO II

Una vez finalizada la partida, aparecerá un diálogo con la puntuación total obtenida y las opciones de reiniciar la partida, salir o compartir el resultado. En caso de superar el récord, también se notificará. A continuación, se muestra la vista:

ILUSTRACIÓN 42: VISTA FINAL PARTIDA MODO DE JUEGO CLÁSICO

1.1 Pulsar sobre el modo de juego “Nivel” (Nivel 1). Se nos mostrará el inicio del juego del nivel 1. Se jugará y cuando termine se nos informará de si ha sido superado o no. Dándonos las opciones de reiniciar el nivel, pasar al nivel siguiente (en caso de haber superado el actual) o salir. A continuación, se muestran vistas similares a las que se mostrarán, dependiendo del nivel.

ILUSTRACIÓN 43: VISTAS MODO DE JUEGO POR NIVELES

1.2 Pulsar sobre el modo de juego “Equipo”. Se nos mostrarán las opciones de “Crear Partida” o “Unirse a Partida”.

ILUSTRACIÓN 44: VISTA CREAR O UNIRSE A PARTIDA MODO DE JUEGO EN EQUIPO

Dependiendo de la opción que escojamos, el proceso será diferente. En un modo tendremos que crear la partida rellenando el nombre de partida y en otro nos deberemos unir a una partida poniendo nuestro nombre de jugador y seleccionando la partida a la que queremos acceder. En las siguientes vistas, se muestran ambos procesos:

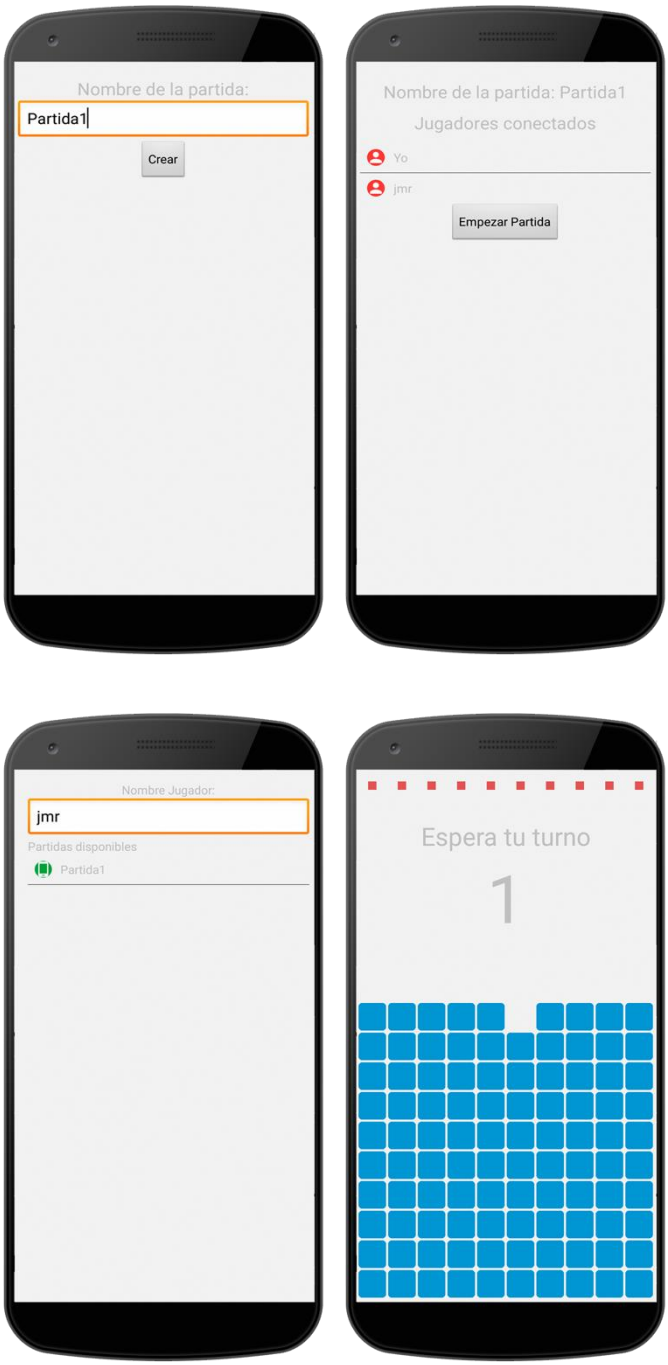

ILUSTRACIÓN 45: VISTA CREAR Y UNIRSE A PARTIDA

Al poner el nombre de la partida y pulsar crear (imagen superior izquierda) nos aparece la vista de la imagen superior derecha. Al principio en “Jugadores conectados” sólo aparecerá “Yo” pero a medida que se vayan uniendo jugadores, irán apareciendo uno debajo del otro.

En la imagen inferior izquierda, aparece la vista que se le mostrará a un usuario que se quiere unir a una partida. Por una parte, deberá rellenar el nombre de jugador y por otra, seleccionar una partida de las de la lista para unirse a ella.

La última vista (imagen inferior derecha) pertenece a la partida ya empezada, cuando un jugador se encuentra esperando su turno.

Nota: en cualquiera de las opciones anteriores nos puede aparecer un diálogo informándonos del funcionamiento del juego. Podemos pulsar sobre “No volver a mostrar” y no volverá a aparecer.

3. Pulsar sobre el botón “Ranking”. Nos llevará a la vista de los marcadores que hay (imagen de la izquierda). A Continuación, podremos pulsar en cualquiera de ellos para ver el ranking (imagen de la derecha).

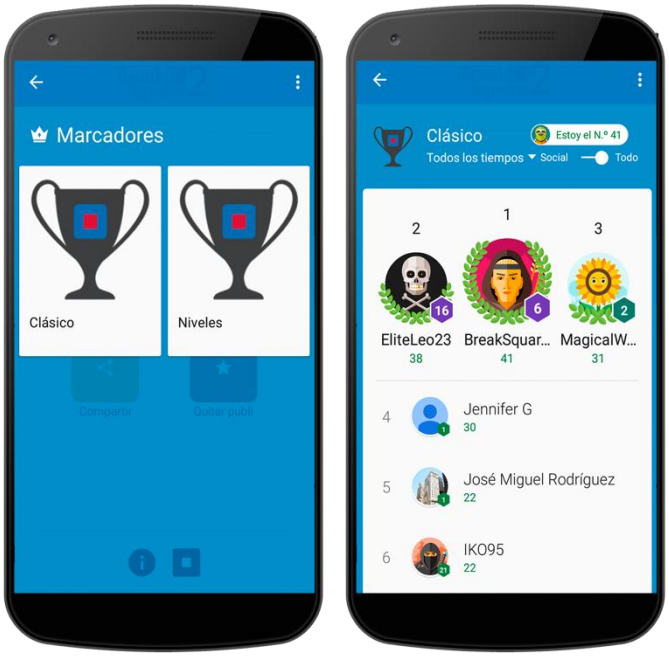

ILUSTRACIÓN 46: VISTA MARCADORES

4. Pulsar sobre “Compartir”. Se nos abrirá una ventana emergente para poder compartir el juego a través de redes sociales, mensajes,

WhatsApp etc. En ella podremos elegir cualquier aplicación en la que compartir el juego.

ILUSTRACIÓN 47: VISTA COMPARTIR JUEGO

5. Pulsar sobre “Quitar publi”. Se nos abrirá un diálogo que nos dará la posibilidad de comprar el producto integrado eliminando así la publicidad. Si se pulsa sobre comprar, el producto se comprará y la publicidad no volverá a aparecer.

ILUSTRACIÓN 48: VISTA COMPRA PRODUCTO INTEGRADO

6. Pulsar sobre el círculo de información “i” de abajo a la izquierda. Se nos abrirá el mismo diálogo que en la imagen nº2.
7. Pulsar sobre el logo de Break Square que está abajo a la derecha ( ) . Se nos abrirá una nueva actividad con información de contacto. También se podrá activar o desactivar el sonido pulsando sobre “SÍ” o “No”.

ILUSTRACIÓN 49: VISTA DE CONTACTO

6.5 POSIBLES MENSAJES DE INFORMACIÓN O DE ERROR

A parte de los diálogos de información de los que hemos hablado anteriormente como pueden ser los diálogos de información sobre el juego o el diálogo para acceder a Google Play Games. En la aplicación existen otros dos mensajes de los que hablamos en este apartado:

1. Mensaje de información para salir de la partida: Dicho mensaje aparece cuando estamos en una partida y queremos salir de ella, para que se nos muestre tenemos que pulsar el botón del dispositivo de “atrás”. El mensaje que se mostraría sería el que se muestra en la siguiente ilustración:

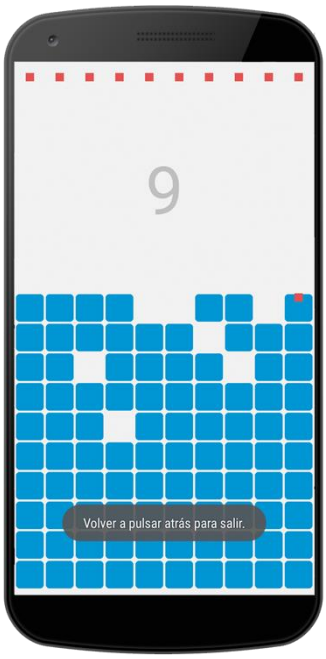

ILUSTRACIÓN 50: VISTA DEL MENSAJE DE SALIR DE LA PARTIDA

2. Mensaje de información para solicitar la activación del Bluetooth: Dicho mensaje, se muestra cuando se accede al modo de partida en equipo, y no está activado el Bluetooth. El objetivo del mensaje es permitir la activación del Bluetooth. Tendremos que aceptar dicha petición para poder jugar al modo de juego en equipo. En la siguiente ilustración se muestra el mensaje:

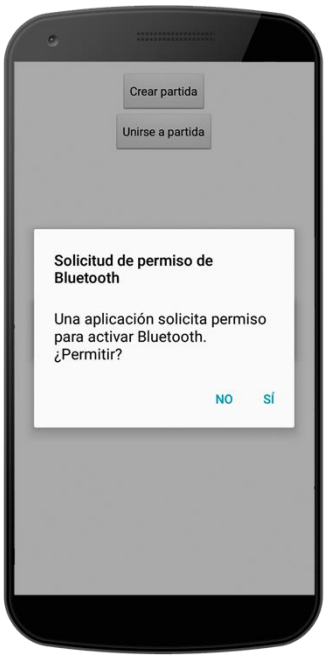

ILUSTRACIÓN 51: VISTA MENSAJE DE ACTIVACIÓN DEL BLUETOOTH

6.6 CONCLUSIONES

Como conclusión de este capítulo, he de decir que también se ha superado de forma exitosa. Se han dado todos los detalles necesarios para que un usuario pueda navegar por toda la interfaz del juego de forma sencilla. También se ha hablado de los requisitos que tiene que cumplir el juego para que sea totalmente funcional.

En el siguiente capítulo, describiré mis conclusiones finales sobre el proyecto y lo que ello ha conllevado.

7 CONCLUSIONES

En este nuevo capítulo, hablaremos sobre las conclusiones a las que se han llegado durante todo el proceso de creación del proyecto, sobre lo que me ha aportado el hacerlo y sobre qué posibles ampliaciones podría tener, todo ello de cara a un futuro.

Se podría decir que en este proyecto se ha partido de 0, porque, aunque tenía desarrollada parte del juego, dicho desarrollo, no se llevó a cabo de la mejor forma posible, ya que antes no tenía los conocimientos que tengo ahora después de haber finalizado la carrera y los he ido aplicando a medida que los he aprendido. Con ello me refiero a la reestructuración del código que he realizado para que sea lo más simple y reutilizable posible, usando técnicas aprendidas en la carrera, como pueden ser la herencia, la POO (programación orientada a objetos) y el MVC (modelo vista controlador). Por otra parte, se ha indagado en los conocimientos que ya tenía a la hora de utilizar ciertas tecnologías, con lo que me he llevado una grata sorpresa al ver detalles en varios aspectos que antes no veía. Por último, también decir que en la parte en la que he tenido que investigar e implementar desde 0 (comunicación) me

he alegrado mucho de haberlo hecho, haber conocido una nueva tecnología que se está expandiendo y haber conseguido los resultados esperados.

7.1 CONSECUCIÓN DE OBJETIVOS

Fueron múltiples los objetivos que se plantearon al principio del proyecto y había incertidumbre por si se podrían cumplir dichos objetivos. Es conveniente verificar que estos objetivos se han llegado a cumplir.

Teniendo en cuenta que los principales objetivos eran crear un juego con los mismos detalles (tecnológicamente hablando) que un juego de una empresa cualquiera podría tener, como podían ser los productos integrados, marcadores, publicidad o varios modos de juego, y, a todo ello sumarle que tenía que consumir la mínima energía posible, viendo el resultado del proyecto, podríamos asegurar que se han cumplido dichos objetivos en mayor o menor medida.

7.2 POSIBLES AMPLIACIONES

Lo primero decir, que la primera ampliación sería pulir varios detalles porque, aunque la aplicación sea completamente funcional, es evidente que se pueden mejorar algunas cosas, como, por ejemplo, la interfaz.

Por otra parte, se podría añadir otro modo de juego multijugador para poder jugar online con una persona que está en otra parte del mundo, como hacen muchos juegos. También añadir nuevos niveles y nuevas ideas que pueden surgir de ello, como poder añadir más productos integrados o nuevas reglas de juego. Por último, sólo quedaría subirla a Google Play y darle publicidad para que fuese descargada.

7.3 APORTACIONES

La realización de este proyecto me ha aportado cosas buenas tanto en lo personal como en lo profesional.

En lo profesional he desarrollado un proyecto como cualquier empresa lo haría en mayor o menor medida, siguiendo paso a paso cada una de las etapas de

desarrollo software (este proceso completo con el mismo software, nunca se ha llevado a cabo en la carrera). Me ha aportado conocimientos sobre una tecnología desconocida como lo era la comunicación bluetooth entre dispositivos. También he ido observando una serie de detalles en los que antes no me fijaba.

Aunque anteriormente ya había desarrollado aplicaciones móviles, eran aplicaciones básicas, por lo que este proyecto también me ha enseñado a que puedo crear aplicaciones más profesionales y con ello me ha aumentado la motivación.

En lo personal me he sentido como un Ingeniero de Software conmigo mismo, tanto a la hora de asignarme mis propias tareas y procedimientos como a la hora de diseñar e implementar la estructura del juego. Gracias a este proyecto, me he dado cuenta que me gusta más la fase de planificación, búsqueda de requisitos y diseño del software que el desarrollo del mismo, lo que hace que me quede más claro lo que tengo que ir buscando a la hora de encontrar trabajo.

En general me siento bastante satisfecho ya que me he demostrado a mí mismo, que puedo abordar un proyecto software desde sus cimientos.

8 BIBLIOGRAFÍA

En este último capítulo, se mostrarán todos los recursos en los que me he apoyado para llevar a cabo el proyecto. Dichos recursos serán; artículos, webs, documentos, etc.

[1] Android Developer Bluetooth, Consultada varias veces desde Agosto de 2016.
https://developer.android.com/guide/topics/connectivity/bluetooth.html?hl=es

[2] Tipos de redes y estándares WIFI, consultada en Septiembre de 2016. https://norfipc.com/redes/tipos-redes-estandares-wi-fi-diferencias.php

[3] AltBeacon GitHub, consultada varias veces desde Septiembre de 2016. https://github.com/AltBeacon/android-beacon-library

[4] AltBeacon, consultada varias veces desde Septiembre de 2016. http://altbeacon.org/

[5] Bluetooth Low Energy, consultada varias veces desde Septiembre de 2016. https://developer.android.com/guide/topics/connectivity/bluetooth-le.html

[6] Understanding the different types of BLE Beacons, consultada varias veces desde Septiembre de 2016. https://developer.mbed.org/blog/entry/BLE-Beacons-URIBeacon-AltBeacons-iBeacon/

[7] EMBC Beacon Packet Specification, consultado varias veces desde Septiembre de 2016. http://www.emmicroelectronic.com/sites/default/files/public/products/datasheets/embc-mn01.pdf

[8] Cannot Detect Alt-beacon in android, consultada en Octubre de 2016. http://stackoverflow.com/questions/37760708/cannot-detect-alt-beacon-in-android

[9] Alt Beacon Android unstable, consultada en Octubre de 2016. http://stackoverflow.com/questions/31172898/alt-beacon-android-unstable

[10] Bluetooth, consultada varias veces a partir de Octubre de 2016. https://www.bluetooth.com/

[11] Facebook, primeros pasos con el SDK para Android, consultada en Octubre de 2016. https://developers.facebook.com/docs/android/getting-started

[12] AltBeacon Protocol Specification v1.0, consultada varias veces desde Octubre de 2016 https://github.com/AltBeacon/spec

[13] Adding Leaderboards to Your Android Game, consultada varias veces desde Octubre de 2016. https://developers.google.com/games/services/android/leaderboards

[14] AdMob, consultada varias veces desde Noviembre de 2016. https://firebase.google.com/docs/admob/

[15] Interstitial Ads, consultada varias veces desde Noviembre de 2016. https://firebase.google.com/docs/admob/android/interstitial

[16] Preparing Your In-app Billing Application, consultada varias veces desde Noviembre de 2016. https://developer.android.com/training/in-app-billing/preparing-iab-app.html

[17] Mobile/Tablet Operating System Market Share, consultada en Enero de 2017. https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=8&qpcustomd=1&qpsp=2016&qpnp=1&qptimeframe=Y

[18] Dashboards, consultada en Enero de 2017. https://developer.android.com/about/dashboards/index.html

[19] Market Brief — Year in Review 2016, consultada en Enero de 2017. https://www.superdataresearch.com/market-data/market-brief-year-in-review/

[20] Informe ditrendia 2016: Mobile en España y en el Mundo, consultada en Enero de 2017. http://www.amic.media/media/files/file_352_1050.pdf