UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Diseño e implementación de videojuegos para dispositivos móviles con el lenguaje Scratch

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Autor: Diego Lara García

Tutor: Xavier Molero Prieto

Curso 2017-2018

Resumen

Esta es la memoria del trabajo de fin de grado titulado “Diseño e implementación de videojuegos para dispositivos móviles en Scratch”. En este trabajo se van a desarrollar tres de los juegos más relevantes de la escena de los videojuegos para dispositivos móviles (2048, Flappy Bird y Fruit Ninja). Para la implementación y desarrollo del trabajo se utilizará el lenguaje de programación Scratch, con tal de demostrar la sencillez, y al mismo tiempo el potencial que puede llegar a tener este lenguaje a la hora de programar videojuegos.

La memoria estará estructurada de tal forma que en primer lugar se hablará de los videojuegos para dispositivos móviles en general y de todo el campo que abarcan, para finalmente explicar qué tres juegos de estos se han escogido para el trabajo y porqué. Seguidamente se hablará de cómo funciona la programación en Scratch y cómo nos podrá ayudar a desarrollar nuestros juegos. Finalmente se les dedicará un capítulo entero a cada uno de los videojuegos escogidos para explicar su estructura y programación en Scratch.

Palabras clave: videojuegos para dispositivos móviles, móviles, diseño de videojuegos, programación, Scratch.

Resum

Aquesta és la memòria del treball de fi de grau titulat "Disseny i implementació de videojocs per a dispositius mòbils en Scratch". En aquest treball es van a desenvolupar tres dels jocs més rellevants de l’escena dels videojocs per a dispositius mòbils (2048, Flappy Bird i Fruit Ninja). Per a la implementació i desenvolupament del treball s’utilitzarà el llenguatge de programació Scratch, per tal de demostrar la senzillesa, i al mateix temps el potencial que pot arribar a tenir aquest llenguatge a l’hora de programar videojocs.

La memòria estarà estructurada de tal manera que en primer lloc es parlarà dels videojocs per a dispositius mòbils en general i de tot el camp que abasten, per finalment explicar què tres jocs d’aquests s’han escollit per al treball i perqué. Seguidament es parlarà de com funciona la programació en Scratch i com ens pot ajudar a desenvolupar els nostres jocs. Finalment se’ls dedicarà un capítol sencer a cada un dels videojocs escollits per explicar la seva estructura i programació en Scratch.

Paraules clau: videojocs per a dispositius mòbils, mòbils, disseny de videojocs, programació, Scratch.

Abstract

This is the memory of the end-of-degree work entitled "Design and implementation of video games for mobile devices in Scratch". In this work, three of the most relevant games of the video game scene for mobile devices will be developed (2048, Flappy Bird and Fruit Ninja). For the implementation and development of the work, the Scratch programming language will be used, in order to

demonstrate the simplicity, and at the same time the potential that this language can have when programming videogames.

The memory will be structured in such a way that in the first place we will talk about video games for mobile devices in general and the whole field they cover, to finally explain which three games of these have been chosen for the work and why. Next we will talk about how programming works in Scratch and how it can help us develop our games. Finally, an entire chapter will be dedicated to each of the selected videogames to explain its structure and programming in Scratch.

Key words: video games for mobile devices, mobile, video game design, programming, Scratch.

Índice general

Índice de figuras

CAPÍTULO 1
Introducción

Este primer capítulo está dedicado a mostrar los motivos por los que me he decidido a realizar este proyecto en concreto. Además, pongo en evidencia los objetivos que pretendo alcanzar, tanto en mi mismo como en el espectador. Uno de los puntos que pertenecen a este capítulo es la breve explicación de como se estructura la totalidad del trabajo, donde se referenciarán, entre otros puntos, las fuentes bibliográficas.

1.1 Motivación

El motivo de realizar este trabajo final de grado (TFG en adelante) es la pasión y el entusiasmo que le he puesto desde pequeño tanto al hecho de jugar los mismos videojuegos como al de intentar entender como era posible que algo que en realidad no existía, y no era físico, pudiera tener una lógica y responder a las señales que emitía el jugador desde el exterior de la pantalla.

Cuando tuve edad para comprender que la funcionalidad que desarrollaba una lógica en el mundo de la informática, y más concretamente en el mundo de los videojuegos, era la programación decidí que mi carrera profesional iba a estar enfocada en este camino. Así pues, con el paso de estos últimos años me he formado y nutrido de los conocimientos para poder realizar tal desempeño en este TFG.

Otra de las motivaciones es demostrar como un lenguaje de programación sencillo como Scratch es capaz de emular la programación de los videojuegos más novedosos ya no solo para las plataformas tradicionales como ordenadores y consolas, sino para teléfonos móviles.

Este TFG lleva un paso más allá de la utilización de Scratch para la docencia en programación, este TFG busca, de forma transgresora y ambiciosa, la programación de videojuegos que hoy en día estamos jugando en nuestros dispositivos móviles y que en ningún caso la gente imaginaría que se puedan gestar con una sola herramienta y una sola persona.

1.2 Objetivos

El principal objetivo de este trabajo es conseguir una evolución personal de programación en el entorno de programación Scratch y mostrar cómo se pueden lograr metas ambiciosas a partir de una herramienta, en principio, sencilla. Para ello se elaborarán tres juegos conocidos actualmente y con alta repercusión en ámbito de los videojuegos para dispositivos móviles.

Para conseguir una recreación lograda se han descargado los juegos y probados para conocerlos en profundidad. Los juegos seleccionados para trabajar sobre ellos han sido 2048, Flappy Bird y Fruit Ninja.

Dejando un poco de lado la idea de programación y la función formativa de este trabajo también se pretende mostrar la repercusión que tienen en la actualidad unos videojuegos que a priori simplemente estaban creados para entretener en momentos puntuales de tu día a día, y como, prácticamente sin esperarlo, se han convertido en grandes apuestas tanto nacionales como internacionales para la creación de eventos y competiciones de carácter profesional.

1.3 Estructura de la memoria

La memoria de este proyecto se divide en 7 capítulos que se explican brevemente a continuación:

- **Capítulo 1:** En este capítulo se muestran, principalmente, la motivación por la que se ha llevado a cabo este trabajo y los objetivos que se pretenden alcanzar una vez esté finalizado. Además, se expone la estructura a seguir durante la redacción y se referencia la bibliografía empleada.

- **Capítulo 2:** En él se muestra una visión actualizada sobre la evolución de los videojuegos en las diferentes plataformas tradicionales hasta alcanzar una plataforma que se utiliza en el día a día, el teléfono móvil. También se hablará sobre el impacto y la repercusión en la industria la aparición de videojuegos para teléfonos móviles, y por último se explicarán brevemente los tres juegos seleccionados para ilustrar el trabajo.

- **Capítulo 3:** En este capítulo se explica el origen del lenguaje Scratch con el que se desarrollan los juegos de este trabajo, la función educativa del programa y las ventajas que nos ofrece a la hora de programar un videojuego desde cero.

- **Capítulos 4, 5 y 6:** Estos capítulos están dedicados a mostrar como se implementa la programación en Scratch de forma concreta para conseguir reproducir cada uno de los videojuegos seleccionados de la manera más fiel respecto al original. Se desarrollarán los juegos 2048, Flappy Bird y Fruit Ninja, tres de los juegos más influyentes del sector de los videojuegos para teléfonos móviles.

- **Capítulo 7:** En este último capítulo se hace una pequeña mirada atrás para recorrer el trabajo realizado y poder observar desde otro punto de vista los objetivos que se han alcanzado.

1.4 Derechos de autor

En este trabajo se han implementado diferentes videojuegos mundialmente conocidos y los cuales, por supuesto, están registrados. Por tanto, este trabajo se ha realizado con fines total y únicamente académicos, para demostrar dentro de las posibilidades de Scratch el potencial que puede llegar a tener este entorno de programación de iniciación a la misma. Todo el material audiovisual de los videojuegos se ha obtenido gratuitamente de las páginas de sprites y sonidos `www.spriters-resource.com/mobile` y `www.sounds-resource.com/mobile/` respectivamente. Así mismo también se han generado algunos sprites e imágenes a partir del juego original y generando algunas modificaciones con la herramienta photoshop para poder adaptarlos a nuestro entorno y a nuestra versión de los diferentes juegos.

CAPÍTULO 2
Los videojuegos en dispositivos móviles

En este capítulo se mostrará un poco más sobre la evolución de los videojuegos hasta alcanzar el nivel de videojuegos plenamente orientados para dispositivos móviles y de las plataformas en las que se han podido disfrutar a lo largo de la historia. También estudiaremos el impacto y repercusión que estos videojuegos han tenido en la propia industria. Para poder adentrarnos más en este intrigante mundo conoceremos tres de los videojuegos más representativos de las nuevas generaciones y de los que todos los aficionados del mundo de los videojuegos han jugado u oído hablar de ellos, estos son el 2048, Flappy Bird y Fruit Ninja.

2.1 Evolución de las plataformas de juego

El principio de la historia de las consolas se remonta al año 1972 con la aparición de la consola Odyssey por la compañía Magnavox y en la que únicamente se podía disfrutar del mítico juego de Pong (2.1) que se limitaba al uso de dos barras verticales a modo de raqueta en la pantalla y un píxel que simbolizaba una pelota.

Figura 2.1: Consola Odyssey por la compañía Magnavox e instantanea de su juego Pong.

Con el transcurso de los años las compañías fueron mejorando sus consolas, dejando algunas plataformas tan representativas como la Nintendo Entertainment System (NES) (2.2) en 1985 de la compañía Nintendo, en la cual se podía

ya empezar a disfrutar de la mítica saga Super Mario Bros o del emblemático Tetris (2.2).

Figura 2.2: Consola Nintendo Entertain-ment System (NES) y dos de sus míticos juegos, Super Mario Bros y Tetris.

Otra de las consolas de la época que marcaron la diferencia fue la Master System en el año 1986 de la compañía Sega en la que se dieron a conocer juegos como la saga traída del cómic Astérix o el inolvidable Sonic (2.3).

En el año 1990, Nintendo se aventura a sacar la primera consola portátil, la Game Boy (2.4), con la que se dispararon las ventas y que con el paso del tiempo irían mejorando en sucesivas versiones. Este es probablemente el punto de la historia en el que se empieza a suplir la necesidad creada por el aficionado a los videojuegos de querer llevar su entretenimiento a cualquier parte. Aquí, en este punto de inflexión,nace la idea en la que se basan los juegos de este TFG, la cual pretende llevar el entretenimiento a la plataforma que día a día transportamos con nosotros mismos.

La gran aparición de las consolas PlayStation (2.5) realizadas por la, aún no conocida en el sector, compañía Sony fue en 1995. Esta gran serie de consolas que hoy en día engloban prácticamente la totalidad del mercado junto con las Xbox, ya implementaban el soporte físico en CD y fueron evolucionando hasta la aparición y lanzamiento de la PlayStation 4 en 2013.

Figura 2.3: Consola Master System de la compañía Sega y dos de sus míticos juegos, Astérix y Sonic.

Por otro lado, Nintendo seguiría avanzando con el mercado de las consolas portátiles de la gama Game Boy (2.6) y absorbiendo la totalidad del mercado de dichas plataformas.

En 2001 dio su aparición al mercado, por parte de la compañía Microsoft, la que sería hasta día de hoy la gran competidora de la famosa PlayStation, la Xbox, la cual iría seguida del lanzamiento consecutivo de todas las Xbox (2.7) que existen hasta día de hoy, coincidiendo su lanzamiento en fechas con las de la plataforma de Sony.

El gran intento de la compañía Nintendo de unirse a las grandes plataformas como PlayStation y Xbox fue en el año 2006, cuando se aventuraron con un nuevo modelo de plataforma con interacción física de movimientos mediante el mando y un sensor óptico de posición, esta consola se llamaría Wii y causaría un gran impacto en el mundo de los videojuegos, pero jamás logró superar las estadísticas de Microsoft y Sony.

Figura 2.4: Primera consola portátil Game Boy.

Figura 2.5: Las 4 generaciones de consolas de PlayStation: 1. PlayStation One (1995), 2. PlayStation 2 (2000), 3. PlayStation 3 (2006), PlayStation 4 (2013)

De forma paralela, durante todos estos años, ha existido una plataforma única que ha ido evolucionando y no ha necesitado renovarse con un nuevo lanzamiento cada cierto tiempo. Esa plataforma en la que los aficionados podían disfrutar de los mismos juegos que los aficionados a las consolas ha sido siempre el ordenador, el Personal Computer (PC). Todas estas plataformas han absorbido durante toda su historia todos los eventos de videojuegos, grandes congregaciones de gente para jugar han protagonizado los torneos de videojuegos profesionales más multitudinarios hasta la fecha. El bombazo ha llegado en estos últimos años en los que se ha revelado contra todo pronóstico una plataforma que nadie esperaba que fuera a resaltar en este mundo. Esta plataforma es el teléfono móvil. Así es, el teléfono móvil ha logrado suplir las necesidades que intentaron suplir las Game Boy, pero a un nivel superior, pues realmente han conseguido que todo el mundo lo utilice y juegue en cualquier momento o hueco de tiempo libre en su día a día. Los videojuegos para teléfono móvil han conseguido un auge tan impresionante que ahora los torneos, competiciones y congregaciones de videojuegos se llegan a hacer para juntar a gente jugando sobre dispositivos móviles. De aquí surge la idea de implementar tres videojuegos para teléfonos móviles en Scratch.

Figura 2.6: 4 de las generaciones de consolas Game Boy: 1.Game Boy Color (1998), 2. Game Boy Advance (2001), 3. Game Boy Advance SP (2003), Nintendo Dual Screen (2005)

2.2 Impacto y repercusión en la industria

El mercado de los videojuegos no ha dejado de crecer en los últimos años y no tiene indicios de que vaya a dejar de hacerlo en los próximos, se ha convertido en una de las industrias más potentes del sector del entretenimiento.

Al principio, cuando la industria del videojuego apareció, muchos la clasificaron como simple entretenimiento. Estudios del año pasado aseguran haber alcanzado los 109.000 millones de dólares gracias a los más de 1200 millones de jugadores alrededor de todo el mundo que deciden conectarse a través de su consola u ordenador de manera habitual.

El sector no muestra indicios de decadencia, sino todo lo contrario, se prevé que el crecimiento ni siquiera se va a ralentizar a corto plazo. Como dato, en España, el gasto que realizamos en videojuegos ronda los 1.500 millones de dólares, tanto en los mismos juegos como en las plataformas que lo sostienen, creciendo cada año un 1,5 por ciento. España se ha convertido en el noveno país del mundo en la estadística de videojuegos y juegos online.

Entrando ya más en profundidad en el mundo de los videojuegos para dispositivos móviles, podemos encontrar datos que confirman que en 2013 aproximadamente el 80 por ciento de los beneficios provenientes de las aplicaciones móviles fueron generados por los juegos destinados a estas pequeñas pantallas.

Dentro de este sector de los videojuegos se ha creado una necesidad en el jugador medio, estos desean que sus dispositivos tengan las características tecnológicas suficientes para soportar el rendimiento exigido por los juegos que cada día demandan mas recursos. Esta necesidad se ha cubierto de forma satisfactoria por las marcas de telefonía, que han sido conscientes de los posibles beneficios que iban a producir este nuevo público al que tenían que satisfacer. La tecnología

Figura 2.7: Las 3 generaciones de consolas Xbox: 1.Xbox (2001), 2. Xbox 360 (2005), 3. Game Boy Advance SP (2014)

de los teléfonos móviles ha mejorado de manera más que notable, beneficiando tanto a consumidores del producto como a la industria productora.

Los juegos para móviles han conseguido algo que es uno de los objetivos más complicados para cualquier empresa o agencia de marketing, que es alcanzar un margen de interesados y de público increíble. Prácticamente cualquier persona con un teléfono inteligente tiene descargado algún tipo de juego descargado. Por lo tanto, la existencia de tanto público ha hecho que la producción de juegos a través de la descarga de una aplicación se multiplique y que existan multitud de tipos y con distintas características de juegos.

Este tipo de videojuego ha hecho olvidar la idea de industria dedicada a gente empleando solamente horas frente a su videoconsola u ordenador, ha hecho que el objetivo de las compañías sea esa gente que está jugando al 2048 mientras espera el metro o jugando al Flappy Bird en la cola del banco.

Otro de los puntos importantes que ha alcanzado la industria de los videojuegos es la competición a nivel profesional. Los llamados e-sports han alcanzado un nivel de despliegue económico y empresarial que podríamos relacionar directamente, salvando las distancias, con el mundo del futbol y de los clubes profesionales de deportes tradicionales.

Las partidas competitivas despiertan tal fanatismo en los aficionados que se crean auténticas retransmisiones profesionales a través de plataformas online en las que hay equipos idénticos a los destinados al sector televisivo, con producción, reporteros, locutores etc. Es por ello por lo que las partidas profesionales pasan a llamarse partidos, como si de un deporte tradicional se tratase.

El año pasado en la final del torneo mundial del juego llamado League of Legends se congregaron más de 40.000 aficionados en directo desde el Estadio Nacional de Pekín, entradas por las que algunos pagaron hasta 1.700 euros de reventa. Los más de 75 millones (75.546.408 espectadores) de personas que si-

guieron el partido lo hicieron desde sus mismas casas a través de las distintas páginas online que tenían permisos de retransmisión, y en el caso concreto de España se habilitaron significativos cines en numerosas ciudades importantes del país para poder verlo en directo.

Es evidente la evolución de los videojuegos hacia alcanzar un carácter competitivo y deportivo, pero todo esto no sería posible sin las grandes apuestas económicas que han hecho numerosos clubes para alcanzar la profesionalización del sector. Existen ya equipos que aparecieron desde el principio de todas las competiciones, pero también existen incorporaciones de clubes de deportes tradicionales que han podido observar el avance y las inversiones, cada vez mas suculentas, como el Valencia C.F. que ha destinado una parte de sus gastos a la formación del equipo Valencia C.F. eSports. Esto es simplemente un indicativo más de hacia donde está avanzando la evolución de los videojuegos.

Y como era de imaginar, los videojuegos de teléfono móvil no han querido quedarse atrás de esta parte de la industria y han conseguido colocarse en numerosos eventos organizando torneos y creando una expectación realmente de infarto hacia algunos de sus juegos más populares que tienen la posibilidad de competición interna y de carácter competitivo como el Clash Royale o el Arena Of Valor.

2.3 Juegos seleccionados

Esta sección se dividirá en 4 subsecciones para explicar de forma detallada cuales son los juegos de referencia del sector de los videojuegos para dispositivos móviles y cuales han sido los seleccionados a desarrollar en este trabajo e implementar en Scratch, 2048, Flappy Bird y Fruit Ninja.

2.3.1. Los juegos referentes en el sector

A nivel competitivo los videojuegos más populares han sido siempre los que cuentan con un gran equipo de desarrollo detrás, y por lo tanto con un gran apoyo económico, como podrían ser juegos como el Clash of Clans o Clash Royale apoyados por la prestigiosa compañía SuperCell. Otro de los juegos más recientes que han aparecido en la escena competitiva es el Arena Of Valor, que ha sido desarrollado de mano de la compañía SuperCell y Riot Games conjuntamente, siendo esta última compañía la creadora del famoso juego de ordenador League of Legends, ya comentado con anterioridad.

Obviamente estos juegos tienen unas mecánicas y necesitan de todo un equipo digno de una empresa para el desarrollo de los mismos. Es por esto por lo que se ha decidido desarrollar tres de los juegos más jugados a nivel ocio por los usuarios. Con unas mecánicas que han sido fácilmente trasladables al entorno de programación Scratch y que emulan casi a la perfección el juego original, y todo esto desarrollado por una sola persona. Estos juegos seleccionados son los ya mencionados 2048, Flappy Bird y Fruit Ninja.

2.3.2. 2048

Llegados a este punto, comenzaremos la aventura hacia la implementación de videojuegos de los últimos años para dispositivos móviles con el lenguaje Scratch. El primer afortunado será el 2048.

En la figura 2.8 podemos observar el propio juego en la pantalla de un teléfono móvil.

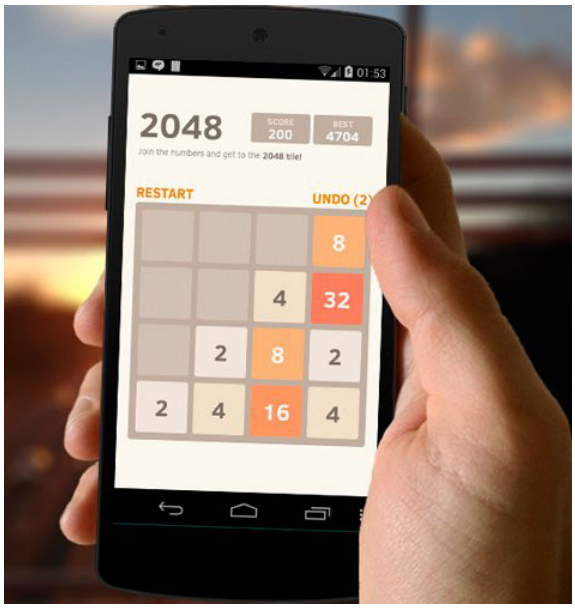

Figura 2.8: Juego 2048 siendo ejecutado en un dispositivo móvil.

Momento de aparición

2048 es un juego rompecabezas lanzado por el joven italiano de 19 años Gabriele Cirulli en 2014.

La idea principal del juego no es de lo más novedosa, puesto que ya fue utilizada en juegos anteriores como el 1024 iOs. Aún así, la revolución causada por este juego fue gracias a las mejoras en mecánicas y la simplificación del concepto, pudiéndole otorgar a esta maravillosa joya del entretenimiento un aspecto más sencillo y de mayor calidad en cuanto a gráficos y animaciones, hechos que facilitarían su propagación por las redes y que captaría la atracción del espectador al verlo por primera vez.

El creador decidió utilizar para su creación HTML, CSS y JavaScript. Pero nosotros hemos decidido llevarlo a un nivel superior de simplificación y lo hemos transportado a la plataforma Scratch contando simplemente con la apariencia física y sonora del juego y conociendo las dinámicas del juego.

Descripción

El objetivo principal del juego es conseguir la consecutiva combinación de números juntos, los cuales serán potencias de 2, con la finalidad de alcanzar el número 2048.

El espacio en el que se desarrolla es una cuadrícula de 4x4 compuesta por 16 espacios cuadrados. El juego comienza con la aparición de 2 "baldosas" que contienen el número 2 con un 90 % o un 4 con un 10 % y tras cada movimiento aparecerán más de estos números de forma aleatoria en los espacios libres de la cuadrícula.

Las mecánicas principales del juego son realizar movimientos verticales y horizontales que desplazarán todas las baldosas existentes hacia ese movimiento hasta fundirse con una baldosa que contenga el mismo número o bloquearse con una con un número diferente. Cuando el jugador consiga que dos baldosas con el mismo número se junten se fundirán para formar un nuevo número que será la siguiente potencia de 2: 2 + 2 = 4, 4 + 4 = 8, 8 + 8 = 16... 1024 + 1024=2048, etc.

El juego originalmente acababa cuando llegabas al 2048, pero actualmente el juego una vez alcanzas el 2048 continua hasta el infinito y por tanto la partida solo puede terminar con la limitación de movimientos causada por un bloqueo de baldosas que no pueden fundirse con ninguna otra del mismo número.

2.3.3. Flappy Bird

Llega el turno de conocer el segundo juego del momento, se trata de Flappy Bird, un simpático y a la vez polémico y efímero juego del creador Nguyen Ha Dong.

Momento de aparición

Flappy Bird apareció en 2013 y su tiempo de vida fue a la vez de asombrosa cuasi efímera.

La aplicación fue desarrollada por el creador vietnamita Nguyen Ha Dong. Meses después de su aparición el juego comenzó a ganar fama en prácticamente días y llego a popularizarse tanto que alcanzó la posición de aplicación mas descargada durante toda la semana. Los creadores de contenido para plataformas multimedia como Youtube no pararon de jugar a este adictivo juego.

Todo tipo de trucos, discusiones y videos relacionados con el juego invadían las redes sociales. Este suceso fue, según algunos expertos causa del diseño simple y "poco original" de los obstáculos, que a su vez tenían un particular parecido a las tuberías del mítico Mario Bros.

La combinación de extrema dificultad y sencillez a la hora de crear este juego fue lo que principalmente le llevo a la fama. En menos de tres días el juego había sido descargado de las plataformas de iOS App Store y Google Play Store por millones de curiosos que se convirtieron en jugadores. Esto ocasionó que el creador del videojuego pasara a ser millonario en cuestión de días a causa de la publicidad que incluía el juego.

Lamentablemente Nguyen Ha Dong con 29 años decidió eliminar el juego de las plataformas de descarga. El creador había llegado a su límite personal de presión y declaró que no lo eliminaba por algunos rumores de plagio a Mario Bros, sino porque el en ningún momento pidió fama ni dinero y el nivel de acoso

que llego a recibir por gente que se había convertido en adicta a través de mails le hicieron replantearse si debía dejar el juego público.

Los teléfonos que consiguieron las prestigiosas descargas se llegaron a revalorizar por tener el juego instalado en cifras de hasta 6 dígitos.

Tras esta historia empezaron a aparecer aplicaciones intentando imitar el juego original y que mantendrían la llama de este ya histórico juego viva.

Descripción

Flappy Bird es un simpático juego en el que la principal misión es mantener a un curioso pájaro con vida. Para ello el jugador debe mantener aleteando al pequeño pájaro mediante toques en la pantalla del teléfono que harán que este se eleve y pueda evitar los obstáculos.

Los obstáculos están formados por tuberías verdes que solo cuentan con un reducido espacio para que el pájaro pueda atravesarlas, y es aquí donde entra la dificultad, pues se debe tener realmente una técnica bien desarrollada para conseguir combinar gravedad, velocidad, tiempo de caída y espacio por el que debe pasar el pájaro para poder atravesar cada uno de los obstáculos.

El juego no tiene fin si consigues constantemente mantener al pájaro con vida, pero si este cae al suelo o impacta contra una de las tuberías el juego finaliza.

El reto y lo que causaba real adicción a los jugadores era el orgullo de poder conseguir una marca superior a las que conseguía el resto y poder publicarlo para que el mundo observara su hazaña.

2.3.4. Fruit Ninja

Para terminar con estos tres juegos mostraremos el último de ellos. En este caso se trata del colorido y refrescante juego de la empresa Halfbrick.

Momento de aparición

Fruit Ninja es un juego creado por la empresa australiana Halfbrick, que fue fundada en 2001 dedicada al desarrollo de videojuegos. Halfbrick es un claro ejemplo de la evolución del sector del entretenimiento multimedia, pues ellos comenzaron orientando sus juegos a plataformas más tradicionales, anteriormente comentadas, como la GameBoy Advance, GameBoy DS y Play Station Portatil. Como se puede observar, ya apostaban por las plataformas portátiles, y fue gracias a ello con lo que consiguieron triunfar.

Este momento de triunfo aparece con el éxito del lanzamiento del juego Fruit Ninja en 2010 para iPhone y iPad, con el cual Halfbrick se corvertiría en uno de los desarrolladores de juegos indie más conocidos en el mundo.

Descripción

El juego original de Fruit Ninja nace de la idea y rumor que se tiene sobre los ninjas, el cual dice que estos odian y rechazan todo tipo de fruta.

Pues bien, Fruit Ninja pretende que el jugador encarne el papel del ninja que ha de enfrentarse y destrozar todas las frutas que crucen en su camino. El dedo del jugador realizará el papel de catana y deberá utilizarlo sobre la pantalla, o en el caso del ordenador utilizar el ratón, para partir todas las frutas que vayan apareciendo en pantalla.

El ninja cuenta con tres "vidas". En el caso en el que alguna de las frutas que han aparecido en pantalla vuelva a caer al vacío sin haber sido cortada se perderá una de estas vidas. Cada una de las frutas que son cortadas suma un punto al marcador. En el juego original, por cada 100 puntos recuperas una vida perdida. Esta es una de las diferencias que tiene el juego de Halfbrick con el simulado en este trabajo, puesto que en nuestro juego jamás podremos recuperar vidas.

En el momento en el que el jugador pierda las tres vidas finaliza la partida con la puntuación que haya logrado hasta el momento. Otra de mecánicas del juego es crear confusión al jugador con la implementación del lanzamiento de bombas entre las frutas, las cuales el jugador debe evitar cortar o de lo contrario perderá directamente la partida.

Otras de las dinámicas extra con las que cuenta el juego original son los distintos modos de juego en los cuales puedes jugar sin miedo a perder porque no salen bombas, o simplemente jugar sin tensión. En nuestro caso hemos creado el juego principal y troncal que hizo famosa a la empresa Halfbrick.

CAPÍTULO 3
Programación en Scratch

En este capítulo vamos a descubrir Scratch (3.1) como entorno de programación. Se conocerán sus orígenes para el desarrollo de aplicaciones. También aprenderemos un poco más sobre su función educativa y la comunidad de usuarios que la frecuentan. Para continuar se mostrarán las distintas partes que componen la interfaz de usuario.

Figura 3.1: Portada del entorno de programación Scratch.

3.1 Origen del entorno de programación

Scratch es un entorno de programación con un lenguaje característico que permite la incorporación al mundo del desarrollo de aplicaciones y programación de cualquier tipo de persona, con mayor o menor experiencia en el ámbito, indistintamente.

Scratch fue desarrollado por el MIT (Massachusetts Institute of Technology) (3.2) con la intención de poder trasladar el mundo de la programación y código a cualquier tipo de público para que consiguieran su comprensión de una forma sencilla y didáctica, sin tener que contar con un alto conocimiento computacional. Este equipo fue liderado en el año 2003 por el norteamericano Mitchel Resnick buscando la simplificación de la programación en bloques simulando el juego infantil de "Lego" para poder integrar en los más jóvenes el interés por la creación y desarrollo.

Figura 3.2: Imagen del MIT

A partir de la versión 2.0 del año 2007 apareció la opción de que la comunidad pudiera compartir abiertamente todas sus creaciones y conocimientos para poder comunicarse, aportar ayudas y mostrar el alcance el programa.

3.2 Función educativa y comunidad de usuarios

Desde el momento de aparición de la herramienta Scratch se ha potenciado el interés social por la enseñanza y la introducción por la programación en centros educativos (3.3). Gracias a la herramienta cada vez es más fácil descubrir un nuevo mundo de posibilidades a alumnos en un proceso educativo como pueda ser la educación secundaria obligatoria. En esta se les puede despertar a los alumnos la chispa que aún no habían descubierto que encendería la llama de su interés y que alumbraría el camino de sus futuros estudios y por lo tanto de su futuro profesional.

Figura 3.3: Imagen de la impartición de docencia a través de Scratch

En los últimos años ya se han realizado estudios con resultados muy positivos en centros escolares, en los que ciertos alumnos de másteres universitarios en etapas de finalización de su carrera han acudido a institutos del país para mostrar y comprobar cuan eficientes son estos métodos de enseñanza y como realmente

existen nuevas formas de aplicar la docencia, así como demostrar que existen asignaturas que no se imparten que pueden, verdaderamente, cambiar la visión de futuro de nuestros pequeños estudiantes.

Por otra parte, también existe una parte de la comunidad no tan joven que busca explotar sus límites y explorar nuevas metas. Para ello se proponen encontrar y conseguir grandes resultados a través de una herramienta tan sencilla como es Scratch. Un ejemplo de ello es este mismo trabajo, donde se han conseguido simular grandes juegos del mercado internacional a través de esta potente herramienta.

3.3 Interfaz de usuario

La interfaz de usuario de la herramienta de programación Scratch se divide en tres zonas principales (imagenX). La parte de la izquierda es el espacio destinado al escenario y los objetos que hemos creado en este. La parte que se encuentra en el centro de la pantalla (3.4) es la zona destinada a los distintos bloques de programación, donde podemos elegir las funciones que arrastraremos a la zona de la derecha, en la cual se ubicará el programa y código que vayamos creando para generar el videojuego.

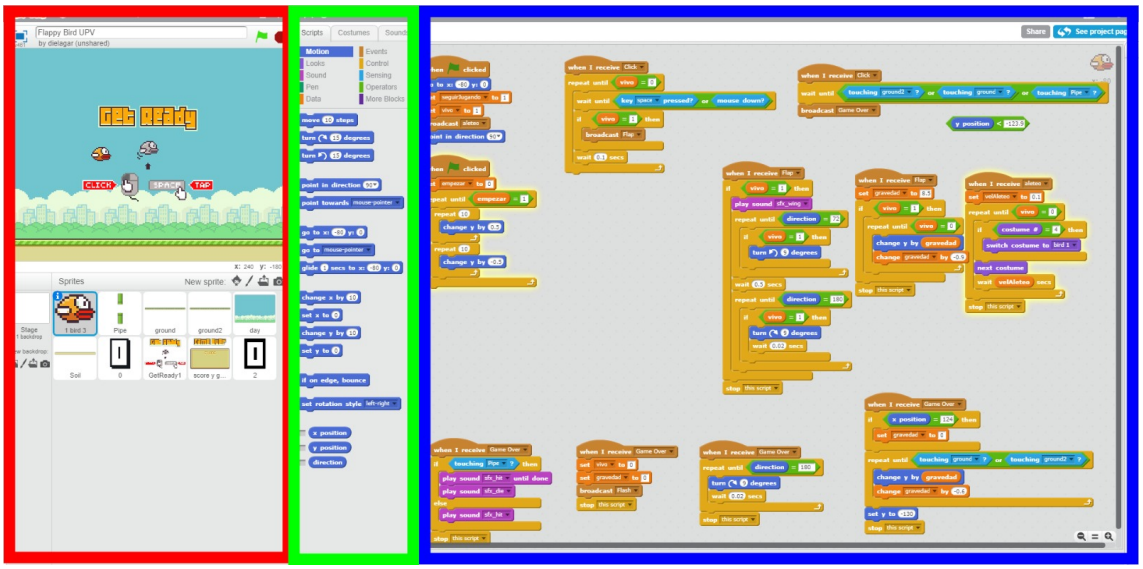

Figura 3.4: Imagen descriptiva de la interfaz de Scratch

3.3.1. Objetos

Los objetos (5.1) son los denominados dentro del entorno de programación como “sprites”. Estos son los cuales contienen las distintas funciones (scripts), disfraces (customes), y sonidos (sounds). Para gestionar cada uno de estos parámetros de cada sprite deberemos hacer click sobre él y tendremos los tres desplegables en la parte de arriba de la zona central.

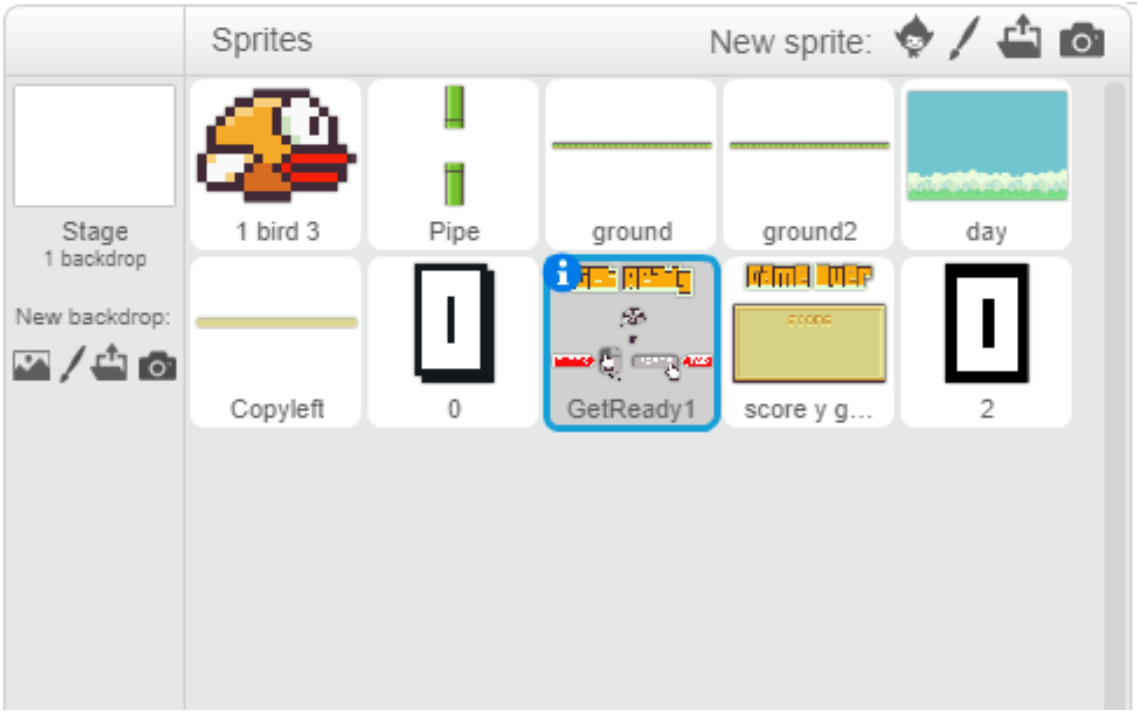

Figura 3.5: Lugar de la interfaz dedicada a los objetos

3.3.2. Funciones

Las funciones (3.6) dentro del entorno de programación Scratch se organizan en distintos bloques:

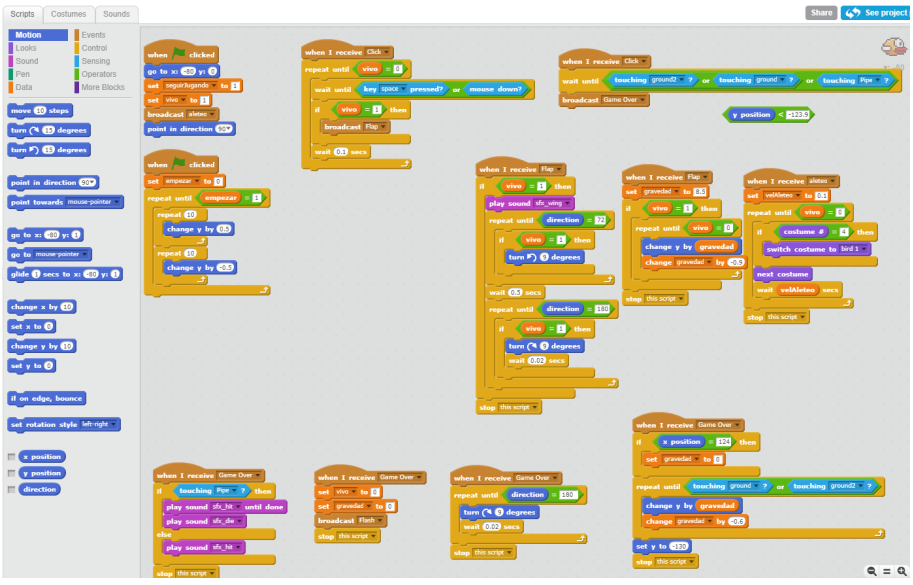

Figura 3.6: Lugar de la interfaz dedicada a las funciones

- **Movimiento:**

  Estos bloques son los utilizados para otorgar propiedades de desplazamiento en la implementación del programa, con él podremos realizar movimientos del personaje o del fondo del videojuego, por ejemplo.

- **Apariencia:**

  Estos bloques son los destinados a los disfraces del objeto. Con él podemos cambiar imágenes de un mismo personaje, ocultarlo, mostrarlo e incluso modificar su tamaño para emular que los objetos generan movimientos como el del Flappy Bird moviendo las alas en el mismo sitio.

- **Sonido:**

  Estos son los bloques empleados para insertar sonidos de los cuales nosotros dispongamos o hayamos obtenido para ambientar el juego en una situación concreta o simplemente para dejar una música de fondo.

- **Lápiz:**

  Se dispone de estos bloques para poder dibujar dentro de la pantalla de juego seleccionando el color, intensidad o tamaño de línea.

- **Datos:**

  Este es un bloque importante en la programación en Scratch puesto que en el definiremos variables locales a un objeto o globales para el conjunto del programa. Estas variables nos darán libertad para crearlas y asignarles valores deseados para el normal funcionamiento del programa.

- **Eventos:**

  Estos son los que se utilizan para hacer responder al programa, a los personajes u objetos. Estos eventos describen cualquier tipo de acción que deba suceder, así como un click del usuario o simplemente una interacción propia del programa que genere un evento en concreto y esto haga desencadenar otra clase de respuestas por causa de este suceso.

- **Control:**

  En estos bloques se nos permite se otorga la opción de reiteración de acciones como en los bucles o los condicionales. También es en estos en los que podremos insertar al programa una espera de tiempo o detener y crear clones de los objetos.

- **Sensores:**

  En este conjunto encontramos la herramienta de controlar la relación que existe entre varios objetos y las posibles interacciones que puedan causarse entre ellos. Nos permite cuando dos objetos entrarán en contacto o cuando pasan por un punto determinado. Otras de las funcionalidades es mostrar el tiempo de transcurso de la partida o controlar si se tiene la webcam habilitada para trabajar con ella.

- **Operadores:**

  Estos son los bloques que abarcan la multitud de operaciones matemáticas que pueden surgir a la hora de programar un videojuego. En estos podemos encontrar desde la función de generar un número aleatorio hasta crear operaciones entre operandos.

- **Más controles:**

  Este bloque permite crear posibles bloques que necesitáramos para nuestro programa y que no están predefinidos en la interfaz estándar de la herramienta de desarrollo. Este es un gran punto a favor del programa Scratch ya que abre una ventana de posibilidades a los usuarios que quieran modificar el límite de alcance del programa.

3.3.3. Disfraces

Los disfraces (3.7) son la representación gráfica a través de la cual podemos percibir los objetos de forma física en la pantalla. Estos los conseguiremos de forma propia y los adjuntaremos como librería de aspectos que puede adquirir el objeto seleccionado y son muchas veces utilizados para dar sensación de movimiento como se ha explicado con anterioridad cuando se han descrito los bloques de apariencias que existen en las funciones.

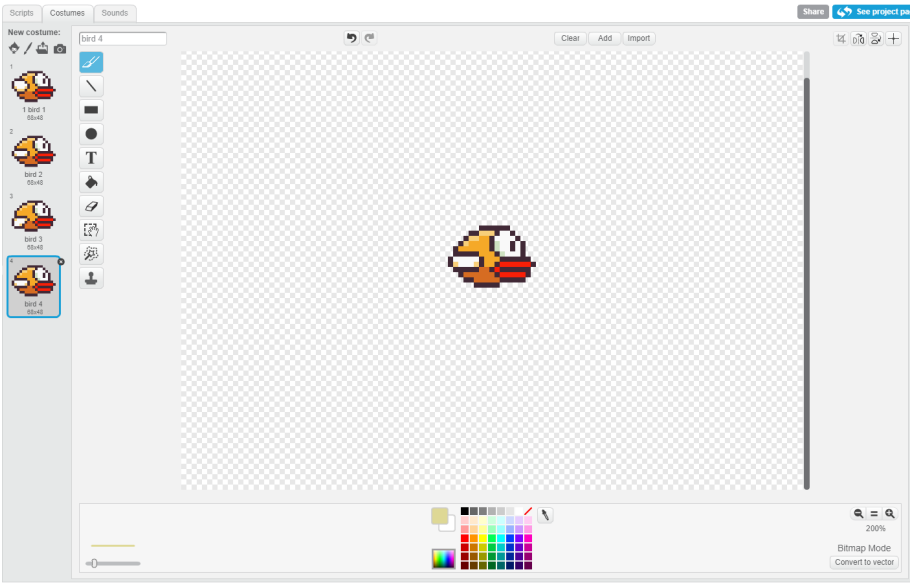

Figura 3.7: Lugar de la interfaz dedicada a los disfraces

3.3.4. Sonidos

Los sonidos (3.8) son los efectos musicales o sonoros que se adjuntan, como ocurre con los disfraces, para formar una librería musical y de la cual se seleccionará el sonido pertinente para el momento adecuado en el cual queremos que ese objeto emita un sonido. Las sintonías se obtendrán de la red o de nuestra propia posesión para, con las funciones disponibles en el bloque de sonido de la zona central de la pantalla implementarlas.

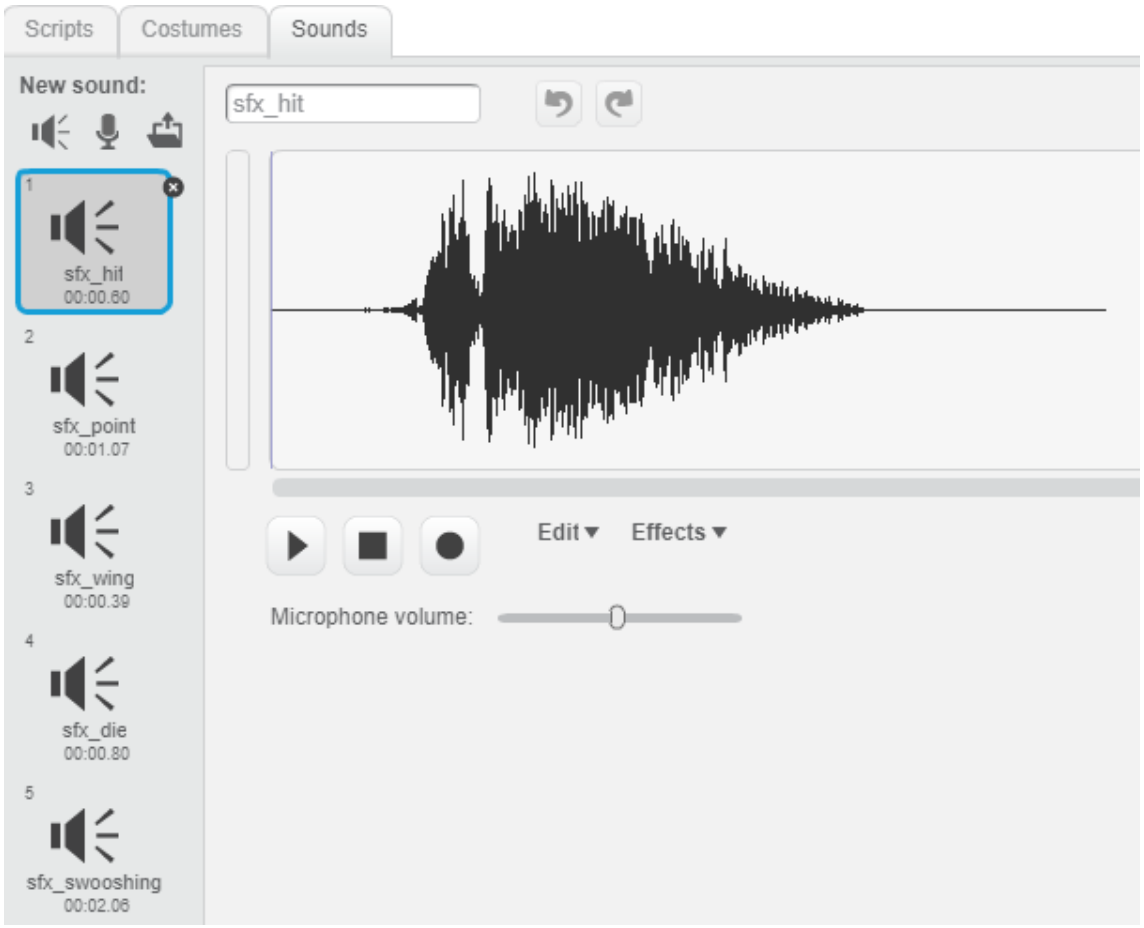

Figura 3.8: Lugar de la interfaz dedicada a los sonidos

CAPÍTULO 4
2048

El juego que se ha realizado en primer lugar en este proyecto ha sido el 2048. El famoso juego matemático de potencias de 2 en el cual hay que desplazar baldosas e ir incrementando sus valores hasta llegar al número 2048 o más. En el capítulo anterior se han mencionado los juegos de más calibre de la escena de los videojuegos para dispositivos móviles, así mismo, también se han mencionado los 3 de los cuales consta este trabajo. En este capítulo se va a tratar específicamente como ha sido el desarrollo del título 2048 en Scratch.

4.1 Toma de contacto

La toma de contacto y probar el juego es lo primero que se debe hacer antes de empezar un trabajo como este, y por tanto en esta sección se tratarán los pasos que se han seguido y lo que se ha realizado antes de empezar a programar para entender mejor todas las partes del juego.

En primera instancia, busqué el juego en Google Play: App Store de Android y me lo instalé en mi proprio dispositivo para probarlo de primera mano, aunque he de decir que ya lo había jugado antes de empezar este trabajo, al igual que el resto de juegos realizados, y ya conocía las mecánicas de este. Gracias a Google Play en Android, la plataforma más grande de subida y descarga de videojuegos, el usuario puede disfrutar en la mayoría de los casos de videojuegos gratuitos para sus dispositivos, dejar su opinion, valorar y recomendar o no cualquier juego.

En definitiva gracias a descargar el juego en Google Play pude probar de forma gratuita y ver cuales eran las mecánicas y cómo funcionaba el juego para tener una idea de por dónde empezar mi trabajo en Scratch.

4.2 Diseño e implementación

En esta sección se hablará sobre el diseño y la implementación del juego que se ha desarrollado en Scratch y en la siguiente sección se tratarán las diferentes variaciones que se han hecho sobre el original.

En primer lugar, hablaremos del diseño y del material utilizado para este videojuego.

En general el juego 2048 es un juego muy sencillo, tanto, que solo necesitaremos la imagen de fondo y las de las baldosas. Todas las imágenes han sido descargadas del juego original, tratadas y adaptadas a Scratch a través de la herramienta Photoshop, generando así sprites en png fácilmente utilizables en Scratch. Principalmente se ha cogido el fondo del juego original y después se han recortado, una a una, cada tipo de baldosa que podía aparecer en el juego (4.1).

Figura 4.1: Visualización del fondo y los sprites de 2048

En el caso del 2048 es un juego sin sonidos por lo que no se ha tenido que descargar ninguno ni añadirlo a Scratch.

Una vez tenemos el material, toca saber cómo distribuirlo y trabajar con él. Para ello se ha diseñado un mapa conceptual para tener una idea de cuales son los eventos y las posibles situaciones en las que se puede encontrar el usuario a la hora de jugar (Imagen mapa conceptual). El programa utilizado para los mapas conceptuales ha sido goconqr (https://www.goconqr.com). Es un programa online muy util para mapas conceptuales, diagramas y cualquier tipo de esquema que se quiera realizar.

En este mapa conceptual 4.2 podemos observar cuales son las simples etapas por las que pasa el juego y cómo cambian en función de las decisiones que tome el jugador.

Al empezar a jugar se generan automáticamente 2 baldosas o bloques los cuales pueden contener o un 2 o un 4 con las probabilidades mencionadas anteriormente. Los bloques se generan en una posición aleatoria del tablero de 4 x 4 sin coincidir en la misma localización. A partir de este punto es cuando realmente comienza el juego para el usuario y comienza su toma de decisiones.

El usuario deberá decidir hacia qué dirección mover y hay 4 posibilidades. Arriba, abajo, izquierda o derecha. Una vez el jugador haya decidido cuál será su movimiento, se comprobará en los cuatro casos para las 16 casillas del tablero, primero si se puede mover hacia el lado indicado, y en segundo lugar si se puede juntar con otra baldosa del mismo tipo y por tanto hacer desaparecer una baldosa y sumar ambas cantidades en la baldosa más cercana al borde de la dirección indicada por el jugador.

Una vez completado este proceso se repetirá de forma iterativa esperando siempre la orden del usuario hasta que de ningún modo ni se pueda realizar un movimiento ni juntar ninguna baldosa con otra. En este momento el juego finalizará y se mostrará la puntuación y la imagen de game over en la pantalla.

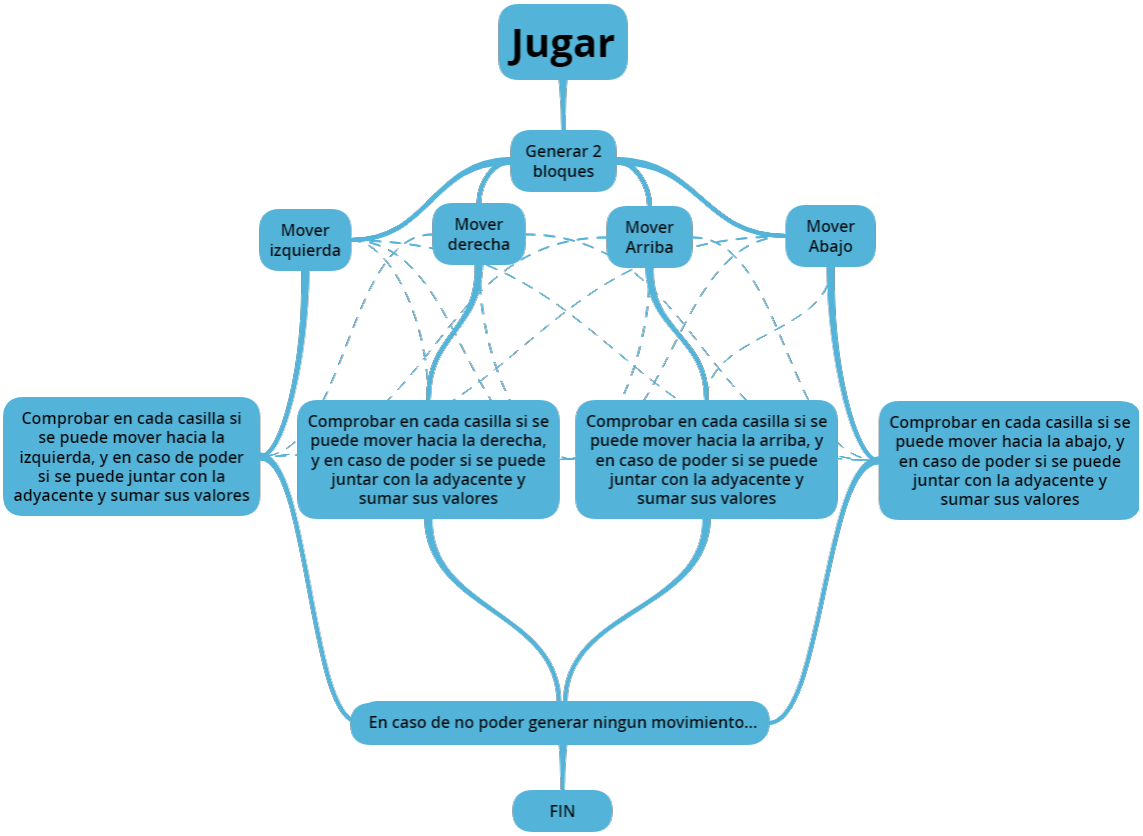

Figura 4.2: Mapa conceptual de los estados y posibles situaciones que se han programado en el videojuego 2048.

Una vez probado, con el material listo y sabiendo cómo abordar las diferentes escenas en las que nos podemos encontrar en el juego procedemos a desarrollar nuestro juego imitando al 2048. A continuación explicaremos las partes del código más relevantes y las cuales nos harán entender el funcionamiento de nuestro juego.

Nuestra versión del 2048 consta tan solo de 2 objetos y el fondo como podemos ver en la imagen (imagen de los objetos y el fondo). El objeto game over tan solo controla que si recibe el broadcast GameOver el objeto vaya al frente, se muestre y haga visible la variable puntuación. Este objeto en caso de volver a empezar se ocultará.

A continuación vamos a pasar a explicar el objeto entorno al cual gira toda nuestra estructura de programación para este juego, el objeto de la baldosa. En él se encuentran los métodos más importantes del título. Este objeto será con el que el usuario interactuará y el cual le dará toda la información necesaria para seguir jugando.

En la imagen 4.3 podemos ver cómo al iniciar el juego se generan diferentes variables. La variable puntuación la cual nos mostrará cuantos puntos llevamos mientras jugamos y al final del juego. La variable moviendo es la que se controlará si se pueden efectuar cambios o no dependiendo de si contiene un 0, lo cual quiere decir que no se está generando ningún movimiento y el programa podrá seguir con normalidad, o un 1, que por lo contrario indicará que sí que se está realizando un movimiento y por tanto no se podrá realizar ningún cambio mien-

Figura 4.3: Inicialización de algunas variables y generación de bloques.

tras esto pase. A continuación se borra toda una lista de valores por si volvemos de alguna partida anterior, no haya conflicto y no se muestre al empezar ninguna baldosa. Seguidamente se genera la lista Valores mediante un bucle que se repite 16 veces y una variable comprobando inicializada a 1, de forma que quede una lista llena de 16 ceros, uno por cada baldosa, estos ceros nos indican el número de la baldosa que se encuentra en esa posición.

Figura 4.4: Posiciones de las baldosas en el tablero de 2048.

En la imagen 4.4 podemos entender la estructura de posicionamiento de cada elemento de la lista, así pues, si en la posición 8 de la lista Valores se encuentra un 32 querrá decir que en la columna 4, fila dos deberá aparecer la baldosa con el número 32.

Para finalizar el evento de inicialización del juego se generan dos bloques llamando al método generaBloque dos veces. Este método se puede observar en la misma imagen. En primer lugar se borran todos elementos de la lista vacíos y

mediante un bucle, con la variable comprobando inicializada de nuevo a uno, se comprueba en la lista Valores elemento a elemento si se encuentra un cero en esa posición y si es así lo añade a la lista vacíos. De esta forma podremos tener en una sola lista las posiciones de todos los elementos que contienen un cero en la lista Valores. A continuación se realiza un pick random de 1 a 100 y si el número es mayor que 90 se reemplaza una de las baldosas que estaban vacías con un 4 en la lista Valores y en caso de que el número aleatorio generado sea inferior a 90 se reemplaza con un 2. Esto se consigue cogiendo un elemento aleatorio de la lista de vacíos que habíamos generado, el cual nos indicará un elemento aleatorio de la tabla y que a su vez se encontrará vacío sí o sí, lo reemplazaremos por el valor que corresponda según el valor obtenido de 1 a 100 del pick random anterior.

Figura 4.5: Imagen como se gestionan en el código los 4 posibles movimientos.

Después de haber iniciado el juego y con ello algunas de sus variables y los métodos que generan los primero cambios en ellas, podríamos decir que el juego queda en espera al usuario para que decida en función de los 2 bloques o baldosas generadas cual será su primer movimiento. Para ello contamos con 4 eventos que pueden que se produzcan, uno por cada dirección que puede elegir el usuario mediante las flechas del teclado (4.5). En el momento que cualquiera de los 4 eventos sea ejecutado, lo cual querrá decir que el jugador ha tomado su decisión y ha generado un movimiento detectaremos cual de las teclas ha sido presionada y entraremos en el evento correspondiente. Seguidamente se llamará al método SeguirJugando el cual se muestra en la imagen (imagen seguirJugando) y que como resumen, comprueba si la partida ha acabado, el usuario se ha quedado bloqueado y por tanto ha perdido o por lo contrario podemos seguir jugando. En la figura (imagen seguirJugando) podemos comprobar y analizar en detalle cómo ha sido este proceso de comprobación. En primer lugar inicializamos la variable seguirJugando a 0 ya que si el programa pasa todas las condiciones y esta variable no es modificada, el valor devuelto será 0 y por tanto querrá decir que la partida ha terminado y que el usuario ha perdido. Se va a recorrer mediante un bucle las 16 posiciones de la lista Valores mediante la variable auxiliar comprobando y teniendo en cuenta en cada incremento de esta, si la variable seguir jugando

está a 1. La primera condición implementada es la más simple de todas. Sí existe un elemento de la lista Valores que contenga un cero quiere decir que hay una casilla vacía y que al menos se puede realizar un movimiento más. En este caso se cambia la variable seguirJugando por uno y la partida continúa. Si en ninguna de las 16 posiciones hay un 0 quiere decir que todas las casillas tienen un bloque o baldosa en ellas con su valor correspondiente. Llegados a este punto para saber si podemos seguir jugando o no debemos comprobar en cada una de las baldosas si sus adyacentes son iguales y por tanto se podrían juntar sus valores en una sola baldosa o por el contrario son diferentes de sus contiguas y la partida ha terminado. Este proceso de comprobación lo hacemos mediante 4 condiciones más, ya que hay que comprobar para cada baldosa, la casilla de su izquierda, derecha, superior e inferior. Obviamente una casilla que está a la izquierda del todo no puede comprobar si es igual a la de su izquierda ya que esta no existe, por tanto, por ejemplo en la condición de comprobar las casillas de la izquierda solo entrara el programa cuando la variable comprobando sea 2, 3, 4, 6, 7, 8, 10, 11, 12, 14, 15 o 16 y lo mismo pasará con el resto de direcciones. En el caso que alguna casilla sea igual a su adyacente se cambiará el valor de seguirJugando a 1, se saldrá del bucle y el usuario podrá seguir disfrutando de su partida.

Una vez comprobado si el usuario puede seguir jugando o no en la figura (imagen teclas direcciones) podemos ver como se hace un broadcast de GameOver en caso de que la variable seguirJugando sea cero o, en caso contrario llama al método que genera el movimiento dependiendo de la tecla de dirección pulsada. Por último vamos a analizar estos 4 métodos que generan los movimientos de las baldosas en las casillas para ello analizaremos por ejemplo el del movimiento hacia la derecha que se muestra en la figura (imagen movimiento derecha). En este método se si la variable moviendo es igual a cero, lo cual quiere decir que no se está realizando ningún otro movimiento mientras tanto, se inicializan las variables moviendo a 1, columna a 4, ya que recorreremos las filas de derecha a izquierda (la forma de recorrer el bucle depende de la dirección hacia la que queramos mover las baldosas en el método) y a la variable comprobando ses le asigna el valor 1. Después de inicializar las variables creamos una copia de la lista Valores en copiaValores. Una vez inicializadas las variables y tener una copia de los elementos que tenemos en la partida en este instante, empiezan los movimientos. Elemento a elemento de la lista Valores comprobamos si tiene alguna casilla vacía en algunas de las baldosas de su derecha. Como hemos dicho empezaremos desde la derecha y recorreremos las casillas hacia la izquierda. Esto quiere decir que si por ejemplo queremos analizar la fila 1 deberemos comprobar el elemento 3 de la lista Valores y podemos encontrarnos en 2 situaciones. La primeras de las situaciones es que si el valor del elemento analizado es diferente de 0 y está situado en una columna inferior a 4, tenga un 0 a su derecha, y en caso de tenerlo se moverá una posición hacia la derecha, lo cual dejará un hueco donde él estaba y dará paso a que cuando analicemos el elemento 2 de Valores encuentre un hueco a su derecha, se mueva también. Esto se repetirá sucesivamente para todas las filas y columnas. La otra de las situaciones que se puede dar es que la posición hacia donde queremos mover nuestro elemento de Valores ya se encuentre otro elemento con un valor, si el valor de este es el mismo que el del que estamos analizando sumaremos sus valores, lo situaremos en la posición del elemento destino al que queríamos llegar, incrementamos el valor de puntuación

en la cantidad del nuevo valor generado en la baldosa y sustituimos el valor del elemento que estábamos analizado por 0 ya que al juntarse ambas en una baldosa, una debe desaparecer. En el caso de que no sea el mismo valor pondremos el valor del elemento que estamos analizando en la posición más cercana al borde o baldosa de la derecha. Una vez intentados hacer todos los movimientos si los valores de copiaValores que habíamos realizado al principio son los mismos de la lista Valores que hemos estado modificando quiere decir que no se ha realizado ningún movimiento y que por tanto en este momento de la partida no se puede mover hacia la derecha. En caso contrario se genera un nuevo bloque mediante el método generarBloque, se actualizan las imágenes de las baldosas correspondientes a sus nuevos valores y se vuelve a poner moviendo a 0.

Una vez vistos los aspectos más importantes de la implementación de mi versión del 2048 podemos entender el funcionamiento interno de este y como repite los mismos procesos y métodos continuamente hasta que la partida finaliza y se muestra el game over en pantalla.

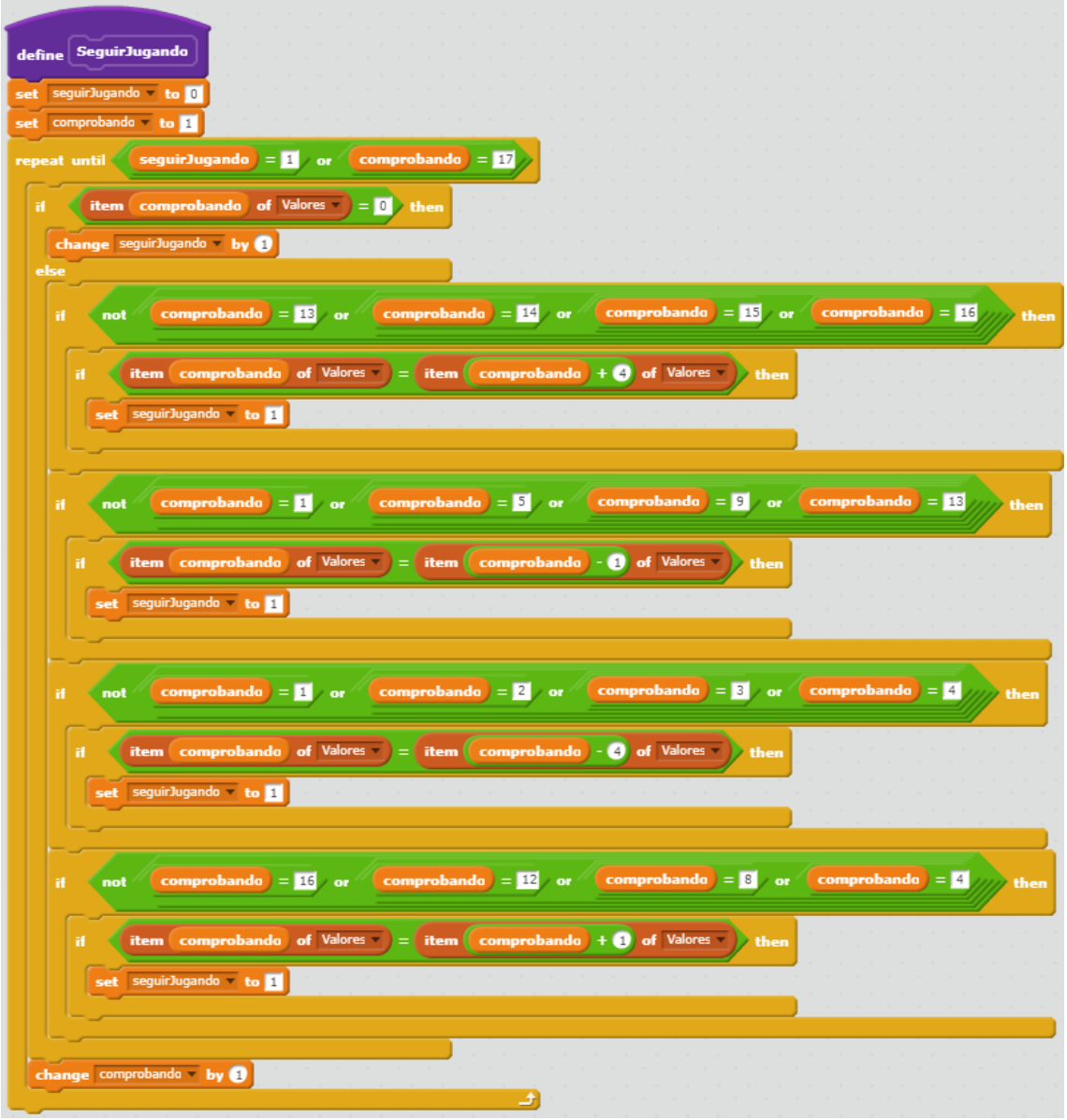

Figura 4.6: Comprobación de la posibilidad de continuar partida tras un movimiento.

(4.7)

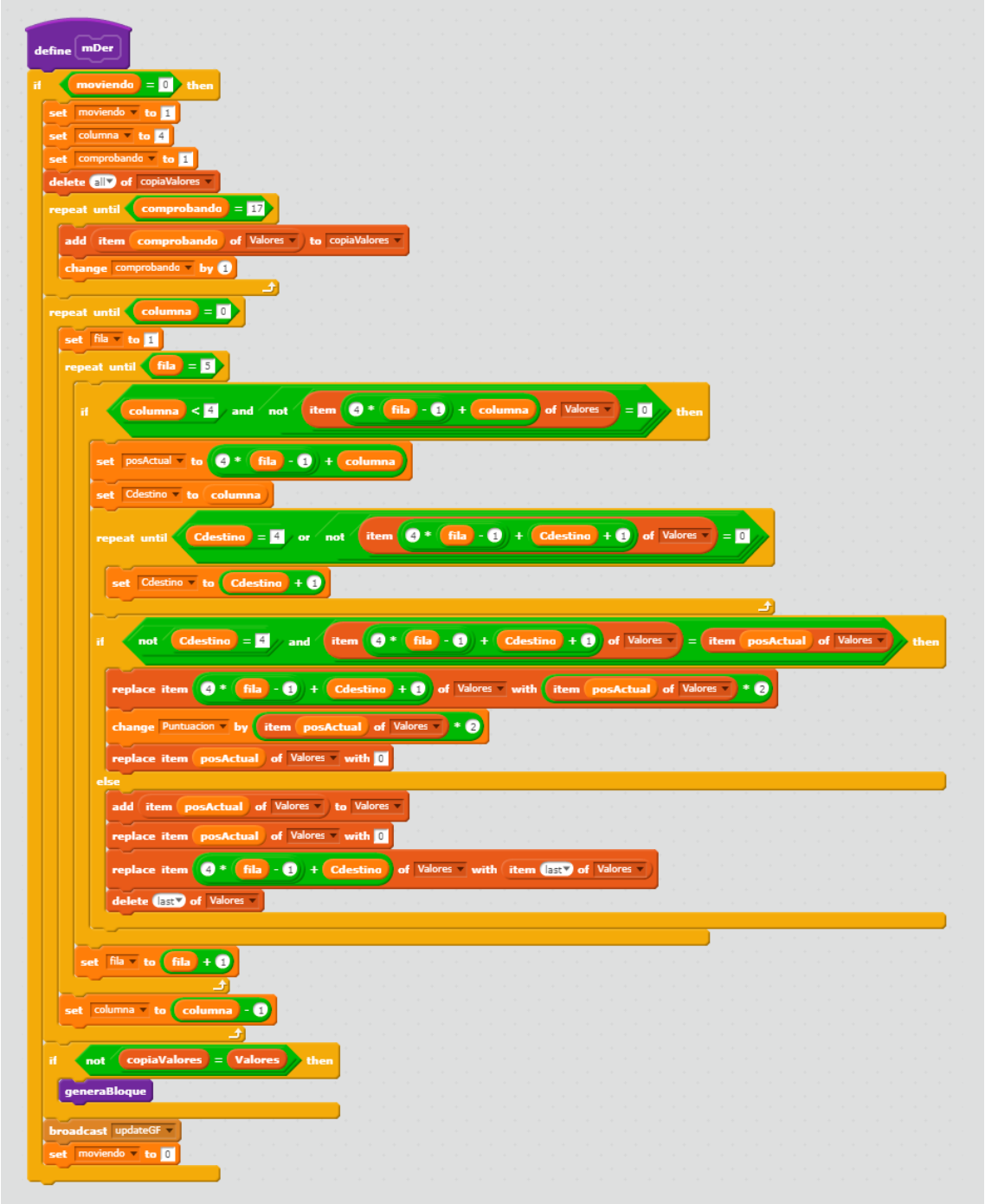

Figura 4.7: Ejemplo de código utilizado para la ejecución de un desplazamiento.

4.3 Aclaraciones

A la hora de programar el juego 2048 han surgido limitaciones y dificultades que han derivado en un resultado no del todo deseado. pues si bien es cierto que el resultado final es lo más aproximado a la realidad, también cabe destacar que uno de los pequeños problemas de programación en Scratch ha sido el intentar

crear una fusión de imágenes al juntar las baldosas. El resultado final ha sido un movimiento inmediato sin progresión, mientras que en el juego original se genera un movimiento visual que finaliza con la fusión de las baldosas. otro de los aspectos que no se han conseguido obtener ha sido la implementación de un marcador particular para el juego, por lo que se ha optado por mostrar el marcador de puntuación que nos ofrece por defecto el entorno de programación Scratch.

CAPÍTULO 5
Flappy Bird

Una vez explicado el juego que más cálculos necesita y el cual requiere una mente ágil y matemática llega el momento de explicar el que podría ser el caso opuesto de videojuego, el Flappy Bird, un juego mucho más dinámico y de mucha habilidad. Este juego que tuvo enganchado a tanto público en el momento de su salida, que tiene como protagonista a un pequeño pájaro con alas desproporcionadamente pequeñas para el tamaño de su cuerpo, y que tendrá que superar los obstáculos que se le irán poniendo en su camino, e incrementar la puntuación en función de como este vaya sobrepasándolos. Para este juego se han seguido los mismo pasos que en el 2048. Primero se ha hecho una toma de contacto con el juego y después se ha diseñado y desarrollado en función del análisis que se ha hecho.

5.1 Toma de contacto

La toma de contacto, al igual que en el anterior título fué descargando la aplicacion a traves de Google Play lo cual resultó igual de fácil que anteriormente. Con una vez el juego descargado e instalado en mi dispositivo movil solo faltaba empezar a trastear, y eso hice.

La primera impresión que te llevas nada más empezar a jugar es que no puedes entender como un juego tan sencillo y a la vez tan difícil se podía hacer viral. Y es precisamente esta dificultad la que te engancha. El sentimiento de no poder pasar ni una tubería el que te lleva a intentarlo más veces, ya que sabes que se puede, porque lo has visto en otras personas. Cuesta mucho adaptarse al juego pero cuando consigue entender más o menos como funciona la gravedad en el pajaro y a la velocidad a la que se mueven las tuberías puedes ir encadenando poco a poco, mas y mas obstáculos superados y por tanto más puntuación. He de reconocer que puede llegar a ser adictivo.

En definitiva, gracias a descargar el juego pude ver la complejidad que se escondía detrás de un juego tan sencillo y disponerme a empezar con mi propia versión.

5.2 Diseño e implementación

En esta sección del capítulo se hablará sobre el diseño e implementación del juego Flappy Bird desarrollado en Scratch y de cuáles han sido las decisiones a tomar para empezar a implementar el título así como también se explicarán en profundidad las partes más importantes de su código.

Al igual que en el capítulo anterior hablaremos primero del diseño y del material que se ha necesitado para este título, y más adelante se hablará de su implementación.

El dinámico juego Flappy Bird tiene algunos objetos más que los que tenía el 2048 ya que aquí hay mas sprites en juego. En este caso a diferencia del anterior las imágenes y los sonidos han sido descargados de los repositorios `http://www.spriters-resource.com` y `http://www.sounds-resource.com` en internet. Como podemos comprobar en la imagen 5.1 necesitamos un objeto pájaro el cual está compuesto de 3 sprites, para las 3 diferentes posiciones de sus alas y conseguir una buena simulación de aleteo. También tenemos un objeto pipe, el cual tiene en total 5 imágenes diferentes con las distintas alturas a las que puede aparecer la tubería. El siguiente objeto que nos encontramos es el césped del suelo, el cual se desplaza, para que parezca que es infinito necesitaremos otro objeto igual el cual aparecerá a continuación del primero para dar este efecto de que el camino no acaba nunca. En este caso una parte del fondo está compuesto por el objeto day junto con otra parte de tierra. De esta forma le daremos contexto al juego. Finalmente también necesitaremos para las puntuaciones los números de la escena que se ejecuta durante el juego y el objeto de la puntuación para la escena final donde se muestra la puntuación alcanzada. Para darle un toque más vistoso también se han añadido un menú de get ready para empezar justo cuando el jugador toque la tecla de aleteo y un fondo para la puntuación final.

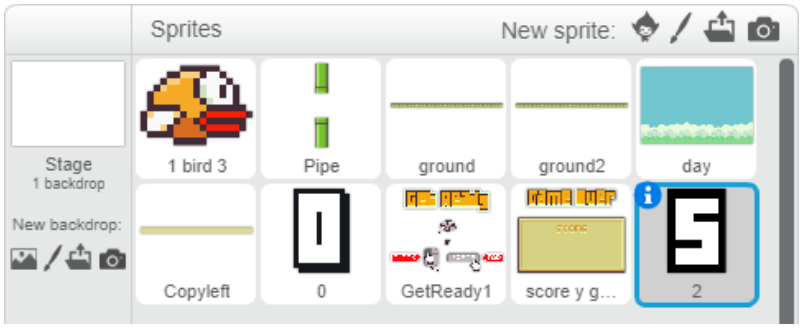

Figura 5.1: Visualización de los objetos de Flappy Bird.

Con lo que respecta a sonidos se han descargado principalmente los sonidos de aleteo, cuando el pájaro choca, el sonido de caída y el de tubería superada.

Con el material a nuestra mano toca de nuevo saber cómo distribuir el trabajo y por donde empezar. Para ello utilizaremos de nuevo un mapa conceptual el cual nos ayudará a entender cómo funcionará nuestra versión del juego mucho mejor. Utilizaremos una vez más goconqr.

En el mapa conceptual del Flappy Bird (5.2) podemos observar por las pocas fases por las que pasa el usuario a la hora de jugar. Es por esta sencillez y por la

Figura 5.2: Mapa conceptual de Flappy Bird.

dificultad de su técnica por lo que este juego rápido se hizo tan adictivo y viral. Las diferentes partes del mapa conceptual empiezan dandole al boton de jugar. el cual nos lleva directamente a la pantalla de juego con el pájaro aleteando sobre el fondo y con el césped ya en movimiento. En esta pantalla se espera a que el usuario accione alguna de las teclas de aleteo ya sea o bien el espacio o el click izquierdo del ratón.

Una vez de comienzo la partida y por tanto el aleteo del pájaro, a los pocos segundos empezarán a aparecer obstáculos (tuberías con diferentes alturas). Esto dará pie a dos posibles situaciones.

La primera situación es que el usuario domine la técnica y consiga superar el obstáculo, lo cual incrementará la puntuación en uno. El usuario debe estar preparado para seguir aleteando y manteniendo el pájaro en vuelo, pues poco tiempo después del primer obstáculo deberá superar el siguiente si quiere seguir incrementando la puntuación.

La otra situación es que el usuario no supere el obstáculo o que toque el suelo y por tanto el pajaro morirá, dejará de aletear y se quedará en el suelo en la misma posición del eje X sobre la cual ha tenido el impacto. Cuando esto suceda acabará la partida y se mostrará la puntuación final en grande en el centro de la pantalla.

Una vez hecha la toma de contacto, el material que necesitamos disponible y sabiendo cómo interpretar el juego gracias al mapa conceptual en el que vemos las diferentes situaciones que nos podemos encontrar llega el momento de desarrollar, implementar en Scratch nuestra versión del Flappy Bird.

En primer lugar analizaremos una parte de la programación del objeto pájaro. Como podemos observar en la figura (5.3) se han implementado 2 dos eventos con diferentes instrucciones cuando se recibe Click, evento recibido del objeto day en el momento en que el ratón se encuentra en la pantalla y se acciona o el espacio o el click izquierdo del ratón. En el primero de estos dos eventos, mientras el pájaro esté vivo, es decir, la variable vivo contenga el valor 1, se esperará a que el espacio o el click izquierdo del ratón sean accionados. En ese momento si vivo sigue a 1 se hará un broadcast de Flap el cual llamará a otras acciones que comentaremos un poco más tarde. El otro evento de Click implementado esperará a que o el pájaro esté tocando el suelo 1, el suelo 2 o un tubo, y en este momento se hará un broadcast de Game Over, el cual hará caer al pájaro en la misma posición de X en la que tuvo el impacto hasta tocar el suelo y sin aletear.

Ahora sí vamos a analizar los eventos Flap implementados. En el primero si el pájaro sigue vivo reproduciremos el sonido del aleteo sfxwing, y acto seguido aumentaremos la inclinación, rotaremos la dirección del objeto hasta 72 de 9 en 9 grados y tras 5 segundos decrementaremos esta inclinación hasta alcanzar en la dirección un máximo de 180, también de 9 en 9 grados con un tiempo de pausa entre cada 9 grados de 0.02 segundos. En el segundo evento de Flap se controla la gravedad. Para ello inicializaremos la variable gravedad a 8.5 y en caso de que el pájaro esté vivo incrementaremos la posición con el valor de gravedad, reduciendo el valor de la gravedad en 0.9 en cada iteración. Este proceso se repetirá hasta que el pájaro muera. Obviamente si la variable gravedad se va decrementando llega un momento en que es negativa y el pájaro cae hasta que se reciba otro Flap.

El último punto a mencionar de esta parte de la programación es el evento aleteo, llamado nada más iniciar el juego para dar sensación de vuelo. En este se inicializa la variable velAleteo a 0.1 la cual indicará a la velocidad a la que cambiarán los disfraces del objeto. Mientras esté vivo se van rotando todos los disfraces del objeto esperando entre cada cambio el valor de velAleteo.

El siguiente objeto que vamos a analizar son las tuberías o el objeto Pipe (5.4). Estos engorrosos obstáculos también tienen su parte de programación para que aparezcan y desaparezcan cada vez con un disfraz aleatorio y se muevan. En primer lugar se inicializa la variable nuevo tubo a 1, variable que nos indicará si hay que generar un tubo o no. Cuando el objeto reciba Click esperara 1.5 segundos y mientras el pájaro este vivo se esperará a que la variable nuevo tubo esté a 1 para crear un clon de sí mismo y cambiar la variable nuevo tubo a 0 ya que no se tendrá que generar otro tubo hasta que este se haya desplazado un poco como ahora comentaremos.

Cuando en este objeto se empiece como clon se cambiará el disfraz aleatoriamente por uno de los 5 tubos que se han subido, pondremos el objeto en la posición de x = 250 e y = 17 y mostraremos el objeto. Seguidamente cambiaremos la posición de x por la variable avance, la cual nos indica a la velocidad que se desplaza el césped y las tuberías. Este proceso lo repetiremos hasta que la posición de x sea inferior a -250. En este momento se borra el clon. Del mismo modo generamos un nuevo tubo poniendo la variable nuevo tubo a 1 en el momento en que la posición del clon sea inferior a 96. También contemplaremos en este objeto el momento en que el pájaro pase por dentro del tubo. Esto lo haremos en el momento en que la posición x del clon alcance el valor -90, es justo en este preciso

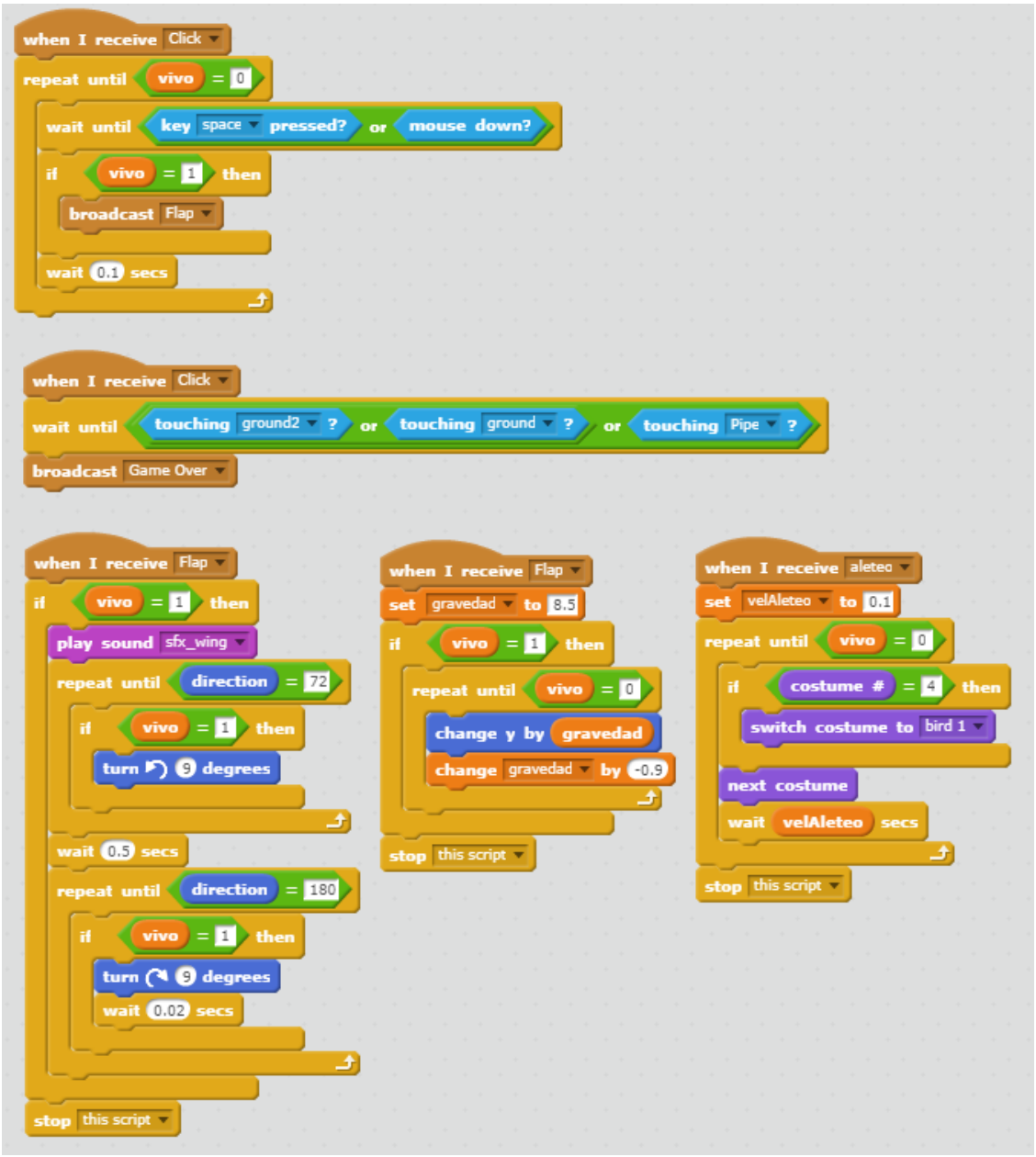

Figura 5.3: Código de la programación del movimiento del pájaro.

instante cuando se reproducirá el sonido sfxpoint, se incrementará en 1 el valor de score, donde almacenamos la puntuación, y el de scoreFinal, variable que se mostrará al final de la partida.

El último, o los últimos objetos analizar son los del césped o ground y ground2. El código de estos dos objetos se relacionan entre sí, es por ello que nos ayudaremos de la imagen (5.5) para entenderlo mejor. En primer lugar ambos objetos nada más iniciarse el juego ponen inicializan vivo a 1. Es en el caso del ground 1 en el que se inicializa la variable Avance a -3.5 y se sitúa este en la posición x = 0 e y = -149. Este se desplazará de forma iterativa en x por el valor de Avance. Paralelamente en el momento que la posición de x sea menor que 0 se llamará a ground 2 mediante el broadcast Suelo 2 el cual sitúa a ground 2 en la posición x = 480 e y = -149 que es la posición más a la derecha de la pantalla. En ground 2

repetiremos el mismo proceso que en ground 1 enviando el broadcast Suelo 1 a ground 1 y cambiando la posición de x a 480 de nuevo.

Gracias a estas partes de código podemos entender como funciona mi versión del juego, como empieza, de qué forma se desarrolla y en qué momento concluye y se acaba. En este caso el juego concluye cuando se ejecuta el broadcast Game Over el cual muestra en pantalla la puntuación sobre la imagen de Game Over.

5.3 Aclaraciones

A la hora de programar el juego Flappy Bird, y a diferencia del 2048 sí que se ha logrado implementar un marcador propio para el juego en el que según vas sobrepasando los obstáculos se va actualizando la puntuación. Por otra parte, el juego original cuenta con el cambio de escenario y de personaje principal, intercalando en el escenario entre el día y la noche y cambiando el color del personaje entre amarillo, rojo y azul. Para simplificar la programación se ha optado por captar la esencia del juego y sus mecánicas dejando un poco de lado la variación física y manteniendo solamente el escenario diurno y el color amarillo del pájaro. En lo relacionado al resto de aspectos se ha conseguido una similitud prácticamente idéntica de tamaños, velocidades, aceleraciones de gravedad, rotación de pájaro etc.

Figura 5.4: Código de la aparición de tubos.

Figura 5.5: Código de la programación del movimiento del césped.

CAPÍTULO 6
Fruit Ninja

Por último y para concluir el proyecto se ha implementado el juego de mayor envergadura de los 3, el Fruit Ninja. En este capítulo hablaremos sobre como se ha desarrollado mi versión de este juego. Empezaremos con la toma de contacto, cuales han sido las sensaciones, el diseño e implementación y acabaremos con unas aclaraciones.

6.1 Toma de contacto

La toma de contacto con este juego ha sido diferente, ya que recordaba como era el juego original de hace años, pero al descargarlo en Google Play el juego era un tanto distinto. Esta vez habia muchos mas modos de juegos, lo cual me asustó. Tuve que tomar la decisión de reducir el volumen del juego y ceñirme solo a la versión clásica.

Por otro lado la impresión general que se puede llevar el usuario que abre por primera vez el juego es una sensación de estar disfrutando de un juego muy dinámico, rápido y entretenido, a la vez que en algunos momentos, algo estresante. Gracias a descargar el juego pude entender que íbamos a tener que lidiar de nuevo con la gravedad, velocidades etc.

En definitiva, sabía que me esperaba el juego mas costoso del proyecto, pero tocaba ponerse manos a la obra.

6.2 Diseño e implementación

Como ya se ha hecho en los capítulos anteriores en esta sección se hablará sobre el diseño y la implementación del juego Fruit Ninja implementado en Scratch. También plantearemos cuáles son las decisiones que se han tomado a la hora de programar y cómo estructurar el desarrollo del proyecto.

Para este juego ha sido más difícil encontrar recursos en internet por lo que he recopilado algunas imágenes y sonidos de otros juegos similares en la comunidad de Scratch. En especial agradecer al usuario quadrupeslap ya que es él quien ha conseguido la versión más parecida y del cual he podido hacerme servir de algunos de sus archivos.

Como podemos comprobar en la imagen (6.1) este juego es el que más objetos requiere, entre los cuales, los más importantes son el puntero, fruit y bomba, los cuales explicaremos más profundamente más adelante.

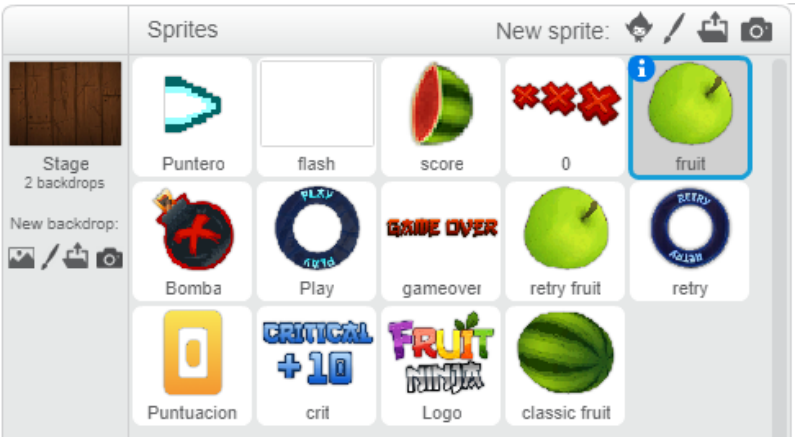

Figura 6.1: Visualización de los objetos de Fruit Ninja.

Con el material conseguido nos encontramos en el mismo punto que en capítulos anteriores. Es el momento de planificar nuestra tarea y saber por dónde empezar. Para ello utilizaremos de nuevo la herramienta goconqr para diseñar un mapa conceptual que nos ayude a entender mejor el camino que seguirá el usuario en nuestra versión del Fruit Ninja.

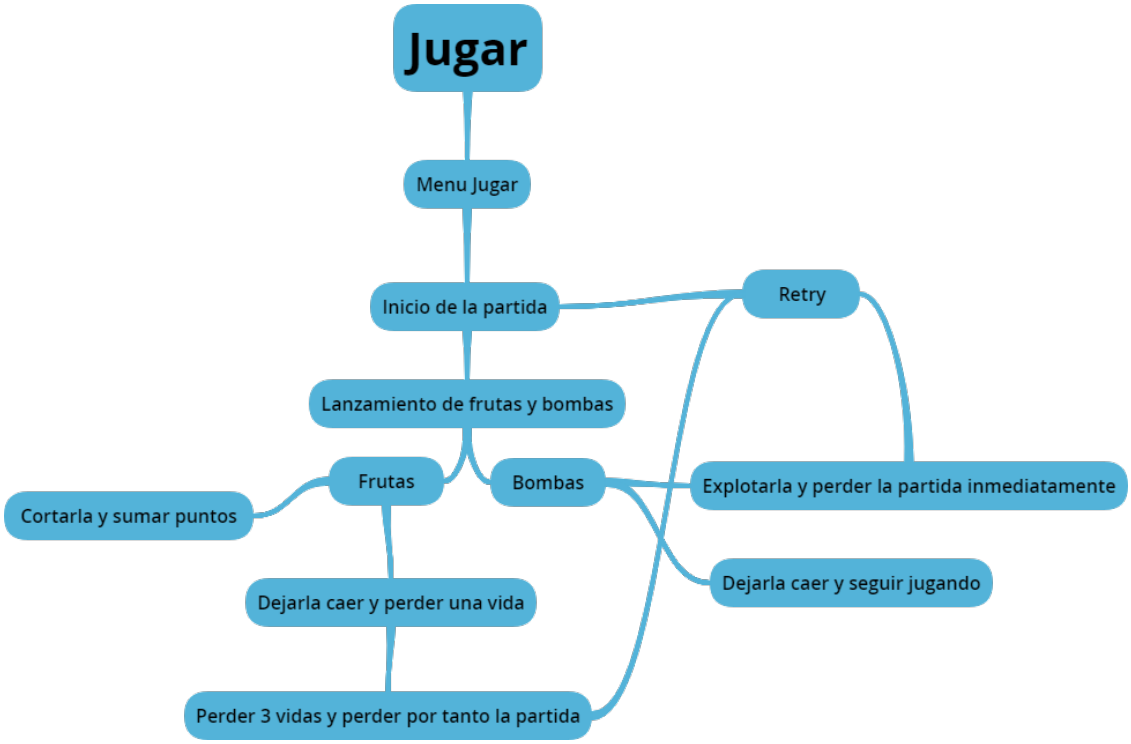

Figura 6.2: Mapa conceptual de Fruit Ninja.

Como podemos observar en el mapa conceptual (6.2). el número de estados por el que puede pasar el usuario es bastante mayor que en los juegos anteriores. Es por ello que era necesario planificarse bien la faena. Gracias a este mapa pu-

dimos entender que el juego necesitaria 3 escenas, la escena menú, la escena de partida y la escena de retry.

En cuanto se abre el juego le aparece al usuario una pantalla con una sandía girando en el menú inicial para ser destruida y comenzar la partida en el momento en el que el usuario desee. Es justo en ese momento, cuando el usuario destruye la sandía, cuando comienza la partida y empiezan a salir frutas y bombas, al principio a una velocidad moderada pero a medida que vamos incrementando nuestra puntuación la velocidad de estas es mayor y por tanto aumenta la posibilidad de que fallemos o destruyamos una bomba. Centrándonos en las frutas, por cada fruta destruida sumaremos un punto, y por cada fruta fallida perderemos una vida. Podremos perder hasta un máximo de 3 vidas. En el momento en que perdemos las 3 vidas acaba la partida y vamos al menú retry donde se nos mostrará la puntuación y se nos dara la opcion de volver a intentarlo. Por otro lado están las bombas, que podemos evitarlas y poder seguir jugando o por lo contrario destruirla y perder directamente la partida, ignorando por completo cuántas vidas nos queden. En el caso de explotar la bomba también iríamos directamente a la escena de Retry donde se mostrará lo anteriormente mencionado.

Una vez hemos tenido la toma de contacto y tenemos a nuestra disposición todo el material podemos comenzar con nuestra tercera andadura en Scratch. Es el momento de desarrollar nuestra versión de Fruit Ninja en Scratch.

El juego consta de muchos objetos con varias partes de código cada uno, es por ello que en esta parte solo analizaremos las partes de código de los objetos más importantes o relevantes para el correcto funcionamiento del juego.

En la imagen (6.3) podemos observar el código de una de las partes más importantes de nuestro juego, el puntero. Para implementar el puntero debemos hacer que la imagen siga al puntero del ratón mediante un point towards mouse-pointer y un go to mouse-pointer.

Para simular un corte continuo se hemos creado clones infinitos del propio objeto para que constantemente se esté dibujando en pantalla y desapareciendo. Para que solo haya un disfraz de punta y el resto sean del cuerpo del puntero la primera vez que se ejecuta se pone el disfraz puntero y el resto de clones tendrán el disfraz cuerpo. El tamaño de estos se reduce en -10 6 veces antes de destruir el clon. Obviamente estos clones y el propio objeto original se mostrarán cuando la variable estacortando, la cual nos indica si tenemos el ratón sobre la pantalla y estamos apretando el click izquierdo, sea igual a 1. Los disfraces se ocultaran por el contrario cuando esta variable sea igual a 0. Lamentablemente la velocidad de ejecución de Scratch no es tan alta como para generar una línea perfecta y se ve una línea seccionada. Sin embargo nos deja intuir perfectamente el efecto de corte.

En la figura (6.4) podemos observar la que posiblemente haya sido la parte más difícil de implementar. Se trata como gestionamos cada fruta o como creamos clones para los trozos de fruta destruidos en el aire. Empezamos la programación de esta parte del código cuando somos llamados como clon en el objeto fruit. En este momento añadimos uno a los elementos que ses encuentran en pantalla y marcamos como 1 la variable esClon para hacer saber al programa que no estamos tratando la el objeto principal. Si el estadoFruta es entera, ponemos la variable cortada a 0 y ponemos el disfraz del tipo de fruta que corresponde con el

Figura 6.3: Código de la programación del comportamiento del puntero.

join Fruit tipoFruta. A continuación esperamos hasta que estacortando sea igual a 1 y nuestro puntero esté tocando la fruta. En el momento que esto suceda se cambia la variable cortada a 1 y se genera un broadcast frutaDestruida, inicializando a continuación las variables clon x y clon y con sus respectivas posiciones. También se inicializa la variable clon rotación con la variable dirección. Todo esto se hace para saber la posición en la que la fruta se ha cortado y en qué dirección estaba mirando para que otros métodos puedan tratar estas variables. Se crean dos clones, uno para cada lado de la fruta destruida, y se cambian los elementos en pantalla restándole 1 ya que al generarse 2 clones más se incrementará en su momento en 2 pero la fruta entera ha desaparecido como tal. En este momento la fruta entera se oculta después de mostrar el disfraz de la mancha correspondiente sobre el fondo y da paso a analizar los 2 trozos de la fruta generados.

Si la fruta es la izquierda, la variable vx se verá asignada con un valor de entre -7 y -1. Se cambiará el disfraz por el del trozo izquierdo correspondiente a la fruta anteriormente cortada. Obviamente después de haber copiado los parámetros de rotación, clon x y clon y, estos son asignados de nuevo al clon de la fruta cortada. Una vez el clon del trozo cae elementosEnPantalla se ve reducido en uno.

Del mismo modo que se ha implementado el trozo izquierdo sigue con el derecho pero cambiando la vx por valores de entre 1 y 7.

Por último en la imagen (6.5) podemos observar en el evento jugar cómo trata las oleadas de fruta en función de la puntuación que obtiene el usuario, dificultando la velocidad del lanzamiento entre oleadas cuanto más alta sea la puntuación.

Por otro lado también podemos ver en esta imagen como se ha implementado el método Lanzar fruta. Comenzamos declarando cuál va a ser la posición de salida de la fruta en una altura de -180 en el eje y un valor aleatorio entre -240 y 240 para el eje x. Si el la posición de x es mayor que 0, es decir, está situado a la derecha del centro de la pantalla, la vx del objeto será negativa tomando un valor de entre -4 y -1. En caso contrario querrá decir que nuestro objeto se encuentra en la mitad izquierda de la pantalla y por tanto le asignaremos un valor entre 1 y 4. En ambos casos le asignaremos una vy de entre 15 y 20. Finalmente le asignaremos una gravedad negativa en función de la vy. Una vez hemos asignado todos los parámetros a nuestras variables mostramos el objeto y pasamos a realizar un bucle. Ejecutaremos este bucle siempre y cuando la posición de y sea menor a -180 y la variable cortada sea igual a 1. Por cada iteración de este bucle cambiaremos la variable x en tantas unidades como el valor de vx nos indique. Del mismo modo se aplicará a vy. La fruta girará en función de vx. Una vez hechos estos cambios la variable vy se verá modificada tantas unidades como el valor de gravedad indique. Esto se repite iterativamente hasta que la fruta cae o es destruida. En el caso de que caiga se generará un broadcast de frutaFallida, se decrementará elementosEnPantalla en 1 y se destruirá el clon.

Gracias a estas partes del código podemos entender mejor el funcionamiento del juego internamente y como se ha implementado en Scratch. Definitivamente este ha sido sin duda el juego más complicado de implementar.

6.3 Aclaraciones

Probablemente la decisión más importante a la hora de realizar la programación del Fruit Ninja ha sido descartar los modos de juego que han ido apareciendo con las actualizaciones de este. Puesto que el juego inicialmente apareció con el modo de juego clásico, es el que se ha decidido implementar de la forma más fiel posible. Dentro de este modo se ha creado una interfaz personalizada en la que el jugador solamente podrá pulsar la opción de jugar, la cual ocasionará que el juego empiece directamente, y al finalizar la partida la interfaz te permite comenzar una partida de nuevo. Una de las principales dificultades que se han encontrado a la hora de desarrollar este juego han sido las particiones de la fruta una vez que son cortadas, puesto que estas se dividen en dos clones y esto creaba conflictos en un principio a la hora de que el programa diferenciara los disfraces apropiados y no variaran sus características de velocidad y aceleración a unas que no les corresponde a una fruta cortada.

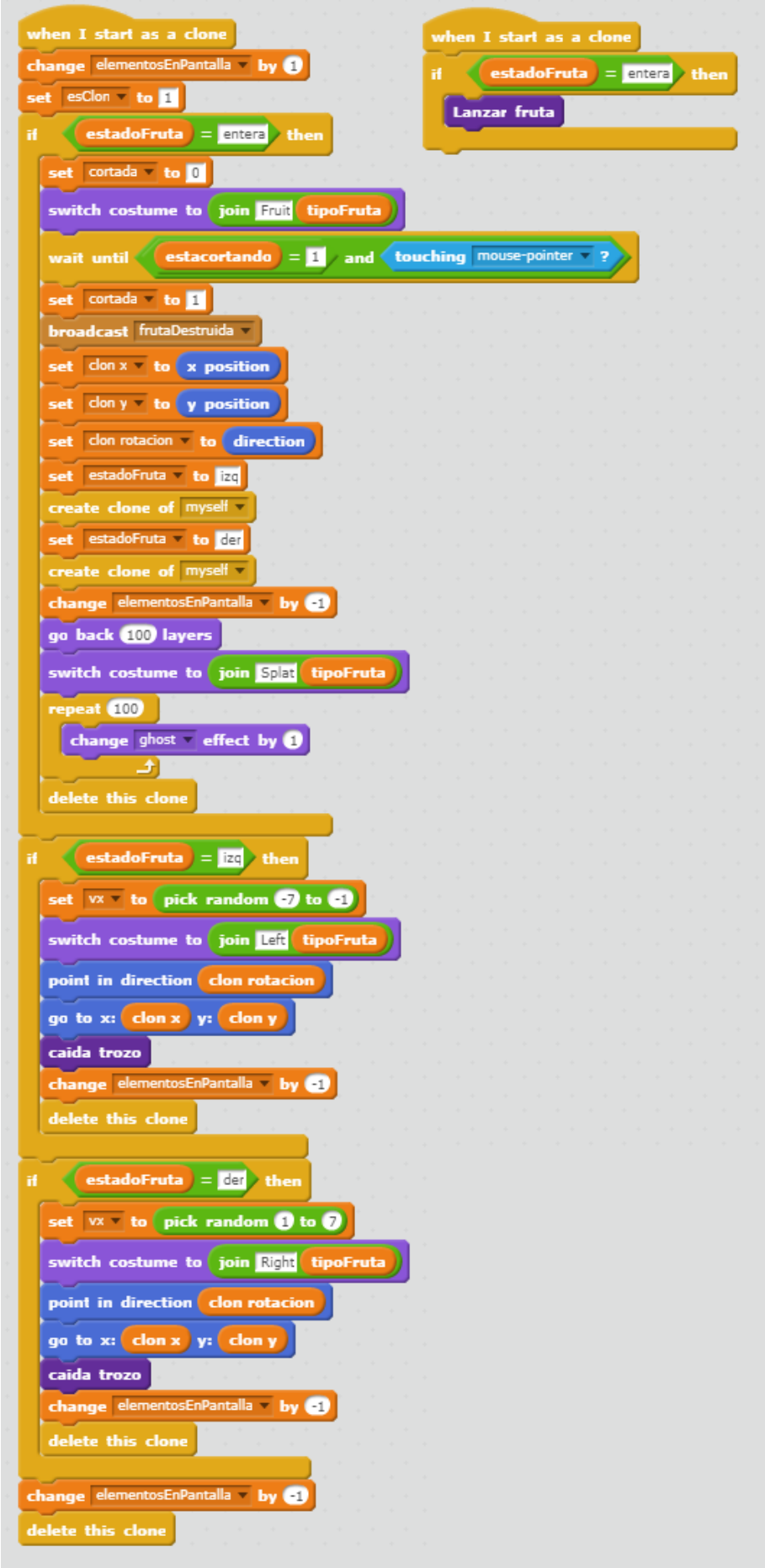

Figura 6.4: Código de la programación del tratamiento de fruta cuando es, o no, cortada.

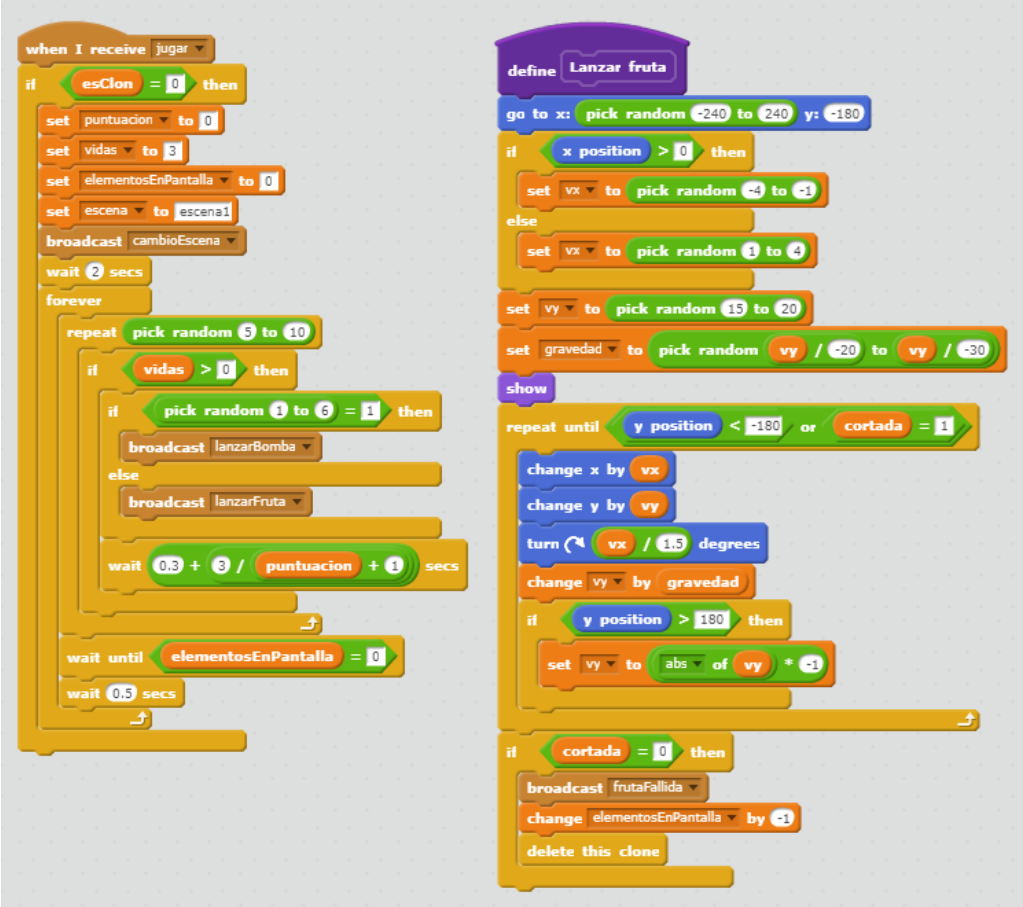

Figura 6.5: Código de la programación del lanzamiento de fruta.

CAPÍTULO 7
Conclusiones

En el presente trabajo se ha estado demostrando y justificando constantemente como un entorno de programación para principiantes y que es de gran ayuda para la docencia puede alcanzar grandes horizontes en el mundo del desarrollo de videojuegos, tal y como lo harían grandes compañías y desarrolladores más profesionalizados.

Basándonos en los artículos, páginas y libros mencionados en la bibliografía se ha conseguido, empleando el lenguaje Scratch los objetivos planteados, alcanzando el nivel de similitud al juego original más próximo con los recursos y tiempo disponibles. Como se ha explicado en el capítulo tres, el entorno de programación Scratch ha sido diseñado para el proceso de aprendizaje de todo tipo de público, en especial el infantil y juvenil. Y es por ello por lo que aun empleando conocimientos avanzados de programación y combinándolos con la programación en bloques de la herramienta, existen metas que no se han podido alcanzar por limitaciones, como es el caso de la implementación gráfica de la catana de Fruit Ninja, la cual no ha sido posible representarla a la perfección. La intención era crear un recorrido continuo, por lo contrario, y por los tiempos de ejecución de Scratch se ha representado de forma intermitente. Estos problemas seguramente se hubieran evitado con otros lenguajes de programación, pero no es menos cierto que también hubiera sido mucho más complejo su desarrollo.

Lo primero que se realiza para poder comenzar con el desarrollo de cualquier juego es una búsqueda de información y de conocimientos de este. En cualquier caso, siempre hemos tenido que buscar el juego original o versiones de este para poder comprender todas las mecánicas, probarlo y explotar al máximo las funcionalidades que podrían ser implementadas en Scratch. A continuación, se describen las distintas partes de la programación realizada para conseguir el resultado final.

Tras el paso de los años y la aparición de actualizaciones de los juegos, estos mismos han ido evolucionando hacia formas más complejas y modos de juego mucho más elaborados. Si bien es cierto que el trabajo principal y la esencia del juego queda reflejada en los tres, para un futuro queda un gran trabajo de actualización y continua adaptación al juego real.

Bibliografía

[1] Escuela Universitaria de Ingeniería de Vitoria-Gasteiz. Departamento de Lenguajes y Sistemas Informáticos. *Manual de Scratch 2*. Emitido el 22 de agosto de 2014. Consultado en http://lsi.vc.ehu.es/pablogn/docencia/FdI/Scratch/Aprenda%20a%20programar%20con%20Scratch%20en%20un%20par%20de%20tardes.pdf.

[2] Wikipedia *2048 (videojuego)*. Emitido el 29 de mayo de 2018. Consultado en https://es.wikipedia.org/wiki/2048_(videojuego).

[3] Periódico digital Gestión. *La fugaz historia de Flappy Bird*. Emitido el 19 de febrero de 2014. Consultado en https://gestion.pe/tecnologia/fugaz-historia-flappy-bird-4417.

[4] myappstand por Marc Ferrandiz *Los ninjas odian la fruta, Fruit Ninja el clásico*. Emitido el 22 de abril de 2012. Consultado en http://www.myappstand.com/los-ninjas-odian-la-fruta-fruit-ninja-el-clasico.

[5] Wikipedia *Fruit Ninja*. Emitido el 1 de abril de 2018. Consultado en https://es.wikipedia.org/wiki/Fruit_Ninja.

[6] *Juego original 2048*. Consultado en https://gabrielecirulli.github.io/2048/.

[7] *Juego original Flappy Bird*. Consultado en https://flappybird.io/.

[8] *Juego original Fruit Ninja para descargar*. Consultado en https://fruitninja.com/.

[9] Varios autores. *Programa tus juegos con Scratch*.

[10] Marco A. Rodríguez. *Programación Visual con Scratch*. 2016.

[11] EST 2015, 18º Concurso de Trabajos Estudiantiles. *"2048 Solution": Algoritmos eficientes para la resolución del juego 2048*. Consultado en http://44jaiio.sadio.org.ar/sites/default/files/est149-166.pdf.

[12] tuxotron *Flappy Bird escrito en 90 líneas de código python*. Emitido el 22 de noviembre de 2015. Consultado en https://www.cyberhades.com/2015/11/22/flappy-bird-escrito-en-90-lineas-de-codigo-python/.

[13] *Crea tus propios juegos sin saber programar con Scratch. Fruit Ninja*. Emitido el 18 de febrero de 2015. Consultado en http://www.chw.net/foro/lenguajes-programacion/1142436-crea-tus-propios-juegos-sin-saber-programar-scratch-fruit-ninja.html.

[14] *Entorno de Trabajo Scratch*. Consultado en http://www.programacionscratch.com/entorno-scratch/.