

Trabajo Fin de Grado de la Universidad de Sevilla



Postmortem de Jack'n Draw

Autor: Alejandro Thomsen Buelga

Curso 2019-2020

ÍNDICE

Introducción	3
1. Descripción del TFG	4
2. Estado de la industria	7
- Estado de las plataformas	7
- Mercado geográfico	8
- La industria del videojuego en España	10
- Andalucía	11
y el problema de los desarrolladores Junior	
- La creación de un marco teórico	12
3. Preproducción	13
- Primera toma de contacto	13
- <i>Change up: La primera propuesta</i>	14
- La propuesta definitiva: <i>Jack'n Draw</i>	17
4. Alpha Dev	19
- Herramientas utilizadas	19
- Elaboración del prototipo	23
- Dinámica de trabajo: Elaboración de cronograma y metodología Scrum	23
- Elaboración del GDD	26
- Primeros artes conceptuales	27
- Programación primeras mecánicas	29
- Problemas humanos, reiteración de código y llegada de la pandemia	30
- Soluciones para la Alpha Dev	31

5. Beta Dev	33
- Enemigos	35
- Multijugador local	36
- Niveles	37
- <i>Power Ups</i>	37
- Jefes Finales	38
- Valoraciones	39
6. QA, <i>wrap up</i> y publicación	40
7. Conclusiones	42
8. Bibliografía	44

Introducción

El siguiente TFG cuenta el desarrollo paso a paso de un videojuego desde el punto de vista de la producción. En este trabajo se analizarán todas las fases de producción del proyecto, empezando por la selección de la idea y el *pitch* interno, la elaboración de los sistemas fundamentales en la *Alpha Dev*, la creación de contenido en *Beta Dev* y finalmente la corrección de errores que se han podido producir en el código, también denominados *bugs*, además de la elaboración de una campaña de *marketing*. En cada fase realizará un ejercicio de autocrítica con el objetivo de que en producciones futuras dichos errores sean solventados con facilidad. Además, dicho trabajo servirá como referencia para los futuros productores de videojuegos que deseen embarcarse en un proyecto; también desea tener una función formativa para quienes solo están familiarizados en el audiovisual tradicional, con objeto de hacerles descubrir que pese a compartir ciertos aspectos, el proceso de trabajo es totalmente distinto. El rol del productor en la industria de los videojuegos suele ser apartado a un segundo plano, lo que hace que muchos desarrolladores junior no tengan en cuenta la dificultad que conlleva producir un videojuego sostenible. Debe tenerse en cuenta, además, el estado de la industria del videojuego español, cuya escasa demanda de trabajo lleva a la mayoría de los jóvenes a emprender mediante una desarrolladora al terminar sus estudios. Por lo tanto, muchos desarrolladores *junior* deciden crear juegos que, debido a una mala planificación o falta de visión empresarial, acarrearán con los siguientes problemas que les impiden ser sostenibles:

- Baja calidad: El juego no está suficientemente pulido en alguno de sus aspectos, lo que redundará en su escaso atractivo y en ventas escasas.
- Inacabable: Muchos jóvenes desarrolladores, no son conscientes de la dificultad que conlleva desarrollar un videojuego desde 0, lo que hace que muchos de los proyectos sean “el juego inacabable” ya sea por falta de personal, experiencia o presupuesto.
- Invendible: Algunos videojuegos, por su mero concepto, no son atractivos para el público. Una mala investigación de mercado, o no establecer un target concreto hará que el juego no tenga un público y por lo tanto, apenas tenga ventas.

Estos problemas suponen un duro golpe de realidad para muchos estudios que han hecho que muchos proyectos sean inacabables, o directamente, si se han acabado y lanzado, apenas han sido rentables, por lo que han tenido que cerrar su estudio relativamente pronto. Además, las ayudas externas que existen apenas han sido de utilidad. Para solucionar estos problemas es necesario el rol del productor. Este se encargará de aquellas tareas de gestión como la organización y planificación del proyecto para que este sea sostenible y rentable a corto y largo plazo. Además se encargará de realizar análisis de mercado y elaborar estrategias de comunicación con objeto de que el proyecto tenga un gran valor para un público concreto.

1: Descripción del TFG

Nuestro proyecto se llama *Jack'n Draw*. Este es un juego de plataformas arcade que hace homenaje al clásico juego *Snow Bros*. En él manejamos a Jack, un joven artista de cómics que debe recuperar su obra combatiendo contra sus antiguas creaciones. En lo mecánico, el núcleo del juego consiste en despejar salas plagadas de enemigos para superar el nivel. Podremos movernos, saltar, lanzar bolas de papel con las que convertiremos a los enemigos en grandes bolas que empujar y lanzar para golpear a otros enemigos, encadenando bajas, y utilizar un aumento de velocidad que nos permitirá ser más ágiles por el escenario. Esto último le confiere más dinamismo y velocidad al juego, elemento diferenciador de *Jack'n Draw* frente a *Snow Bros*. Todo esto podremos jugarlo de manera individual o con un amigo en un cooperativo local.

¿Por qué lo desarrollamos?

Tuvimos varias razones por las que elegimos hacer *Jack'n Draw*, pero fundamentalmente seguimos el consejo de Mauricio García.

Tu primer juego tiene que ser el mínimo producto viable, porque la tentación inicial es hacer el nuevo WoW. [...] Y va más allá. Una persona que está empezando es especialmente incapaz de evaluar lo complicado de un proyecto. La única manera de solventar ese sesgo es copiando un juego existente que además sea muy sencillo (Requena Molina, 2020, p. 163).

En resumen, los puntos a partir de los cuales desarrollamos el proyecto son:

- 1) Referente fijo: *Snow Bros* es un videojuego clásico, que además ha sido ampliamente analizado y estudiado. Esto nos facilita la labor de crear nuevo contenido inspirado, a lo que hay que añadir que se parte de un público concreto.
- 2) Diseño de niveles por *Tile set*: Al ser un juego 2D desarrollado en el motor gráfico *Unity*, nos permite utilizar una metodología de diseño llamada diseño por *tiles*. Estos son elementos prefabricados que se introducen en una paleta de elementos. Gracias a esto podemos “pintar niveles” aumentando la velocidad de producción respecto a la creación de niveles.
- 3) Estilo artístico: El estilo de nuestro referente es muy *cartoon* y nuestro equipo de artistas está muy cómodo dibujando en este estilo en concreto. Esto aumenta la velocidad y la calidad de producción, además de favorecer la comodidad del equipo en el proyecto.

Estructura

Siguiendo paralelamente la producción de *Jack'n Draw*, este trabajo analizará los siguientes puntos. En ellos se contará qué se hizo, cuáles fueron los problemas que se presentaron y cómo se solucionaron.

Idea y planificación (Diciembre y Enero)

Aquí se hablará del proceso que hubo para acabar seleccionando nuestro proyecto. Se explicarán las razones por las que se eligió *Jack'n Draw* como el juego que desarrollaremos durante el siguiente periodo y se mostrará el proceso de creación del *pitch* interno, presentación creada para el propio grupo en la que se explicará brevemente en qué consistirá nuestro juego y cuáles son las razones por las que nos embarcamos en su desarrollo.

***Alpha Dev* (Febrero-Marzo)**

A partir de aquí el juego empezó a producirse. Se mostrará la metodología de trabajo *Scrum* con la que se trabajó en todos los ámbitos del proyecto. Aquí se creará el GDD, documento fundamental en el desarrollo de un videojuego, además de la creación de los sistemas, es decir, el diseño de las mecánicas principales de juego además del *game loop* de juego y el núcleo de este.

Beta Dev (Abril-Mayo)

Una vez establecidos los sistemas básicos, se empezará a crear los contenidos. Esto es todo lo relacionado con niveles, enemigos, mejoras y posible narrativa del juego. Además se empezará a crear el plan de *marketing* del lanzamiento de juego que se pondrá en práctica en Junio.

Quality Assurance “QA” (Junio)

Una vez creado todo el contenido en *Beta Dev*, vendrá la fase de pulido. En ella se corregirán todos los fallos informáticos o *bugs* que tendrá el proyecto. Además, se pondrá en práctica el plan de *marketing* construido en la *Beta Dev*. Finalmente se publicará el videojuego el 23 de Junio de 2020.

Objetivos

Jack'n Draw surge como un proyecto final del curso, por lo que su objetivo primordial es publicar el juego el 21 de junio, independientemente de los demás objetivos que se hayan propuesto. Al fin y al cabo, queremos cumplir con dos de los tres requisitos para que un juego sea sostenible: que sea divertido y acabable. No buscamos ningún beneficio económico, por lo que no necesitamos que el proyecto sea vendible. El segundo objetivo a cumplir es que el juego sea lo más profesional posible. Por muy poco contenido que tenga, este debe estar pulido: tener el menor número de *bugs* posibles y una jugabilidad fluida y divertida. Finalmente, se realizará una pequeña campaña de *marketing* en la que se tratará de crear una pequeña comunidad de al menos 50 jugadores. Este último paso es totalmente prescindible, pero puede servir para analizar la vendibilidad del proyecto y analizar si este realmente es atractivo para algún público.

Finalmente, el objetivo de este TFG es puramente formativo. Con independencia del éxito o fracaso del proyecto, lo que espero es que algún joven productor pueda leer estas memorias para intentar aprovechar todo aquello que hago bien y evite todos aquellos fallos que se cometan.

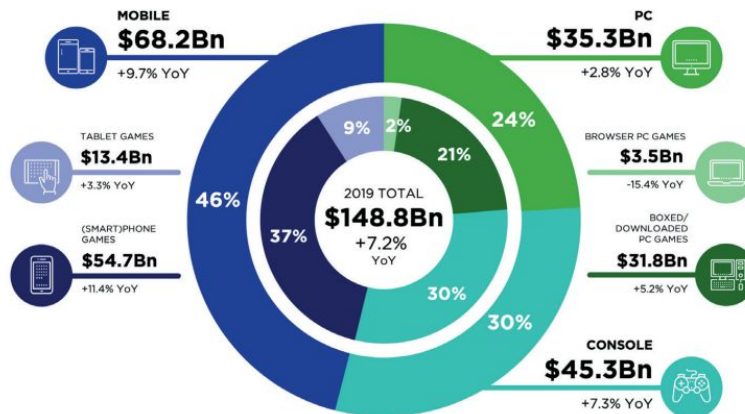
2: Estado de la industria

Estado de las plataformas

Según datos de 2019, la industria del videojuego generó ese año ingresos de 152 miles de millones de dólares, presentando un aumento del 9,6% respecto a los años anteriores. El sector principalmente responsable de esto es el de los videojuegos para móviles, que produce un 45% de la industria global y que en total ha generado 68,5 miles de millones de dólares, lo que convierte al móvil en la plataforma que produce más ingresos. Además, esta sigue en crecimiento eclipsando lentamente a las otras plataformas. En el mercado de consolas observamos una transición de generaciones, en la que las consolas de Sony y Microsoft, PlayStation 4 y Xbox One, dejan paso a las próximas máquinas de ambas compañías, la PlayStation 5 y la Xbox Series X. Mientras tanto, Nintendo asienta sin problemas su consola: la Nintendo Switch. En total han generado 47,9 miles de millones de dólares y se prevé que en 2022 estos ingresos aumentan un 9,7% generando 61,1 miles de millones de dólares. En el PC, el sector ha sufrido una bajada, sobre todo en el caso de los videojuegos para navegador, que empiezan a desaparecer sustituidos por los juegos de móvil. Aun así el mercado del PC sigue estando en alza, con un ingreso de 32,2 miles de millones en 2019, lo que convierte esta en la tercera plataforma que más genera. La mayor parte de los ingresos provienen de micro transacciones de videojuegos que utilizan el modelo de negocio de “videojuego como servicio” (Wijman, 2019).

Se estima que los dispositivos de consola han alcanzado su pico más alto gracias a la consumación de este modelo, según *Global Game Market Report*. Sony anunció en el primer trimestre de 2019 que por primera vez la venta de juegos digitales había superado la venta de juegos físicos, una tendencia que previsiblemente terminará de consolidarse en los próximos años (VVAA, 2020, p.19).

El mercado mundial del videojuego

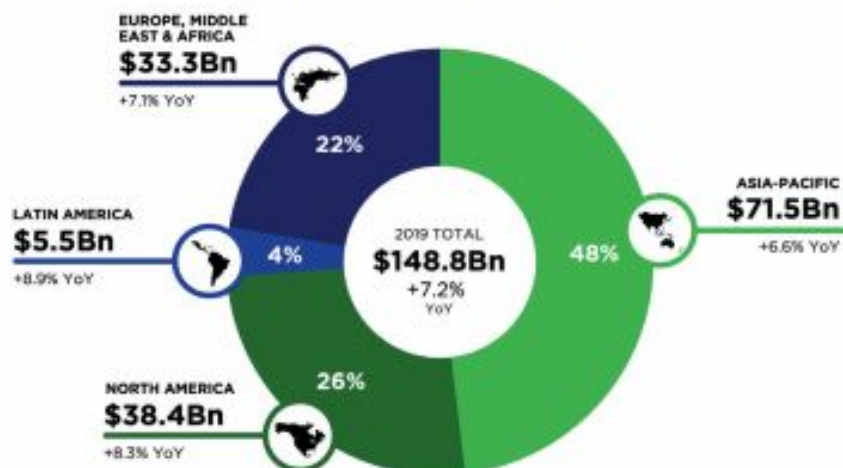


Fuente: Newzoo

Mercado Geográfico

El año pasado, la región de Asia fue la responsable de producir casi la mitad de los ingresos de la industria, recaudando 72,2 miles de millones de dólares, lo que supone un 47% de los ingresos globales. Su crecimiento respecto a 2018 ha sido de un 7,6%. La segunda región que más ingresos ha generado ha sido Norteamérica, con 39,6 miles de millones de dólares suponiendo un 26% de los ingresos globales. Su crecimiento ha sido de un 11,7%. Europa y África le siguen por muy poco con una entrada de 34,7 miles de millones de dólares representando un 23% de los ingresos globales. Su crecimiento ha sido de un 11,5%. Finalmente Latinoamérica se encuentra bastante por detrás de las demás regiones con unos ingresos de 5,6 miles de millones de dólares suponiendo sólo un 4% de los ingresos globales. Aun así, esta sigue creciendo, con un aumento del 11,1%.

Principales mercados geográficos



Fuente: Newzoo

Con estos datos podemos ver que China domina el mercado, acompañado por otros países como Estados Unidos, Japón o Corea. Algo muy remarcable es que España es el noveno país que más videojuegos consume (VVAA, 2020, p. 20). ¿Cómo es su mercado?

Principales 10 mercados mundiales en 2018

#	País	Región	Población (millones)	Tamaño del mercado (M\$)
1	China	Asia	1.420,1	36.540
2	Estados Unidos	Norteamérica	329,1	35.510
3	Japón	Asia	126,9	18.683
4	República de Corea	Asia	51,3	6.194
5	Alemania	Europa occidental	82,4	5.721
6	Reino Unido	Europa occidental	67,0	5.348
7	Francia	Europa occidental	65,5	3.875
8	Canadá	Norteamérica	37,3	2.900
9	España	Europa occidental	46,4	2.583
10	Italia	Europa occidental	59,2	2.547

Andalucía

El esfuerzo de las instituciones y administraciones locales han conseguido que se sitúe como la tercera comunidad autónoma en importancia en la industria del videojuego. La razón de esto se debe en su mayoría al Polo Digital de Málaga que además de ofrecer incubación y acompañamiento especializado para estudios y empresas de videojuego proporcionan apoyo en la búsqueda de financiación y en la atracción de inversión nacional e internacional, además del apoyo de las administraciones andaluzas. El estudio que más empleo genera y que más alcance internacional tiene es Genera Games. Además hay intentos para crear un ecosistema local. Por ejemplo en Málaga se encuentra Gamepolis, que lleva reuniones a todo el sector andaluz desde hace siete años (VVAA, 2020, p. 27), mientras que en Sevilla hay pequeñas iniciativas como DevContact, un evento mensual e informal en el que muchos desarrolladores se reúnen para compartir información.

Los problemas de la juventud andaluza

La juventud de la industria española se puede conocer en su ecosistema. Aquí vemos pequeños estudios operando que no están constituidos como empresas: son pequeños equipos generalmente vinculados a jóvenes que han acabado un grado o un posgrado universitario, que se unen para hacer un videojuego sin crear en paralelo una estructura empresarial. Por lo general, este equipo no suele contar con el rol del productor, lo que hace que muchos equipos no tengan visión empresarial. Esto hace que, por ejemplo, ignoren la posibilidad de obtener financiación por medio de ayudas económicas. Esta inexperiencia y falta de consolidación responde a los bajos ingresos que tienen las empresas. Un 61% factura menos de 200.000 euros al año. Apenas un 3% de las empresas supera los 10 millones de euros al año, aunque es ese 3% el que más ingresos aporta al sector, ya que son responsables de un 67% de la facturación anual.

La creación de un marco teórico

Al ser necesario priorizar el departamento de producción en el desarrollo de un videojuego, es importante empezar a pensar en crear material para que futuros desarrolladores junior tengan referencias para sus próximas producciones. Uno de los materiales más valiosos a la hora de estudiar la producción de un videojuego son los *Postmortems*. Estas herramientas son extensos artículos (como este mismo) en la que se realiza un análisis sobre todos los aciertos y errores que se hayan producido para que futuros desarrolladores puedan leerlos y evitar dichos problemas. Gran cantidad de videojuegos conocidos de la industria AAA han publicado sus *postmortems* en Gamasutra, blog fundamental para el estudio de cualquier perspectiva en el desarrollo de un videojuego aunque también se han publicado en otros medios como Gameindustry o Kotaku. Respecto a documentos oficiales contamos con *Newzoo* que analiza anualmente el estado de la industria en todo el mundo. También debemos tener en cuenta la construcción de un formato audiovisual. El canal de Youtube Game Maker's Toolkit profundiza en la producción de videojuegos y en sus demás aspectos con gran profesionalidad además de organizar todos los años la Game Maker's Toolkit Jam, una de las *game jam* más grandes que existen, reuniendo a miles de desarrolladores todos los años.

En España se está empezando poco a poco a desarrollar un estudio exhaustivo del funcionamiento de la industria e iniciativas para empezar a reforzar la producción de videojuegos. El mayor exponente que tenemos es *El Libro Blanco del Desarrollo Español de Videojuegos* que anualmente, al igual que *Newzoo* realiza un repaso sobre el estado mundial de la industria y, sobre todo también analiza el estado de la industria en cada comunidad autónoma del país. Editoriales como Héroes de Papel o Síntesis empiezan a publicar libros que recogen experiencias de producciones de videojuegos como *Sangre, Sudor y Píxeles* o *El modelo europeo de desarrollo de videojuegos*, que tienen gran valor en el estudio de la producción de videojuegos.

Por último me gustaría mencionar a la prensa de videojuegos española con *Manual* revista bi-anual editada en Sevilla que en varios artículos empieza a recoger múltiples aspectos de la

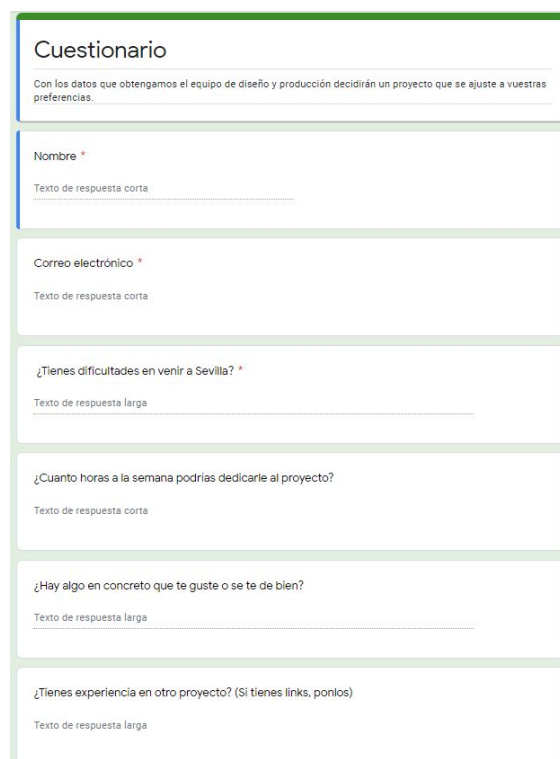
producción de videojuegos tratando de acercarse a lo que hacen revistas internacionales como *Kotaku*.

3: Preproducción

Primera toma de contacto

El 19 de noviembre de 2019 empezó el proyecto con la reunión de presentación de todos los integrantes que desarrollarían el proyecto durante los siguientes meses. Por motivos de confidencialidad no se utilizarán sus nombres, aunque la distribución es de dos diseñadores, dos programadores, tres artistas 2D y un productor. Tras la presentación, agrupé a todos en dos redes de comunicación: WhatsApp y Discord donde expondríamos de forma oral y escrita para tener las primeras reuniones. Además, los de diseño, producción y arte tendríamos nuestras primeras reuniones los jueves a las 12:30, al acabar las clases.

Además, como primera toma de contacto, se creó el siguiente cuestionario. Con los datos obtenidos, seríamos conscientes del tiempo que podíamos invertir en el proyecto, además de si existían dificultades para acudir a las reuniones presenciales, incluyendo finalmente preferencias a la hora de desarrollar el juego.



Cuestionario

Con los datos que obtengamos el equipo de diseño y producción decidirán un proyecto que se ajuste a vuestras preferencias.

Nombre *

Texto de respuesta corta

Correo electrónico *

Texto de respuesta corta

¿Tienes dificultades en venir a Sevilla? *

Texto de respuesta larga

¿Cuanto horas a la semana podrías dedicarle al proyecto?

Texto de respuesta corta

¿Hay algo en concreto que te guste o se te de bien?

Texto de respuesta larga

¿Tienes experiencia en otro proyecto? (Si tienes links, ponlos)

Texto de respuesta larga

La idea fue útil, pues nos permitió tener información valiosa para organizar el proyecto. Por ejemplo, aprendimos que los programadores tenían mucha experiencia como programadores, además de que ya tenían experiencia con anteriores proyectos, aunque solo podrían trabajar durante los fines de semana. Recopilando los datos de todos los departamentos, empezamos a reunirnos para generar una tormenta de ideas a partir de la cual seleccionar aquellas que podríamos desarrollar.

Change up: La primera propuesta

Lo primero que teníamos que hacer era seleccionar una idea, un proyecto que convenciera a todos los integrantes del equipo, que tuviese calidad, fuese viable y contase con un nicho contrastado. Aquella debía ser sencilla y con un reducido tiempo de programación. Se hizo una tormenta de ideas en la que participaron todos los miembros del equipo. Las propuestas eran diversas, pero fueron las aventuras narrativas *point and click* las que más convencieron a nuestro equipo pues en este tipo de juegos, los contenidos con que cuenta el juego importan más que las mecánicas. La hipótesis era que al contar con sistemas tan simples, podríamos centrarnos en la creación de contenidos y así poder presentar un proyecto de calidad a la *Gamelab*. Dicho proyecto sería un juego de horror psicológico en el que trataríamos de superar nuestros traumas a través de la resolución de puzzles mientras que aprendíamos sobre la vida del protagonista a través de los escenarios. Con esta propuesta de proyecto decidimos realizar el primer *pitch* interno en donde estableceríamos en qué iba a consistir nuestro juego, el público objetivo y las razones por las que trabajaríamos en este juego.

Pitching

¿De qué trata nuestro juego?

Es un *point and click* en el que nos pondremos en la piel de un protagonista que tratará de superar sus traumas y conflictos personales intentando salir de una

habitación. Lo que nos diferencia de otros *point and click* es la diversidad de estilos artísticos que habrá en el juego, pues cada zona contará con un estilo propio.

¿Cuál será su público?

Si te gustaron *Fran Bow*, *To the moon* y juegos de este estilo, seguro que te interesará. Además trata de un tema muy presente entre los jóvenes a día de hoy. En una época en la que la depresión y la ansiedad están tan presentes a día de hoy, un juego así es necesario.

¿Cuales son las *key features*?

- Supera tus conflictos a través de múltiples habitaciones en las que deberás buscar la forma de salir encontrando tu reflejo
- Arte 2D *pixel art* con diferentes estilos artísticos
- Narrativa enfatizada en el *worldbuilding* más que en sus mecánicas
- Diseñado con un objetivo terapéutico, buscando concienciar a los jugadores de un problema tan tabú como los trastornos mentales.

¿Cuáles son sus atractivos o magnets?

- *Forgotten Branch*: Actualmente, el género de las aventuras gráficas *point and click* está prácticamente extinto y hay un público que todavía demanda este tipo de juegos
- Arte atractivo: El arte tan variado puede ser un atractivo, pues el juego no se hará aburrido
- Símbolo de una causa: El juego trata temas que están presentes a día de hoy, que demandan ser atendidos y visibilizados.

¿Cuál es su público objetivo?

Jugadores que busquen nuevos *point and click*, les interese el horror psicológico o busquen escuchar a otras personas que padezcan problemas similares a los suyos.

¿Por qué era viable?

Además de que la programación no requiere mucho trabajo, el diseño de niveles basado en escenarios estáticos está pensado para los artistas, dando mucho espacio libre para expresarse como quieran. Además, las animaciones podrían ser simples, lo que nos ahorraría mucho trabajo.

La presentación estaba pensada e incluso habíamos creado un calendario con las fechas en las que se establecía el proyecto. Aunque la realidad nos mostró que nuestras expectativas eran demasiado irreales. Esto hizo que al presentar el *pitching* a nuestro tutor fuese rechazado inmediatamente, recibiendo un duro golpe de realidad. Las razones por las que el proyecto fue rechazado fueron las siguientes:

- A) *Anti Magnets* muy dañinos: Un equipo compuesto por estudiantes, en donde se cometerá fallos producidos por la falta de visión, experiencia y profesionalidad en una obra que toca temas tan conflictivos, son cuestiones que pueden suponer un rechazo inmediato para el jugador y una pérdida de credibilidad para el juego.
- B) Elaboración de muchas animaciones: Por lo general, los *point and click* actuales cuentan con una gran cantidad de animaciones. Si nosotros no somos capaces de llegar a ese mismo nivel o simplificamos mucho las animaciones, esto puede suponer un rechazo automático por parte del jugador, pues el juego no se acerca a sus competidores.
- C) Elaboración de un guion: Pese a que algunos miembros del equipo tienen experiencia como guionistas, realizar el guion de un videojuego es algo totalmente diferente, por lo que podría llevarnos una cantidad de tiempo considerable que podríamos invertir en otras tareas,
- D) Dificultad de diseño de un *point and click*: Una cosa es programar un *point and click*, pero diseñarlo y hacer que sea divertido es algo totalmente distinto. Teniendo en cuenta todos los referentes, diseñar un *point and click* que sea divertido y no sea una copia mediocre de algo que ya existe es algo que supera nuestras capacidades actuales.

Con esta respuesta negativa, se tuvo que dar un paso atrás y volver a la selección de ideas. Lo que se nos recomendó fue buscar un juego arcade que tuviese mucho tiempo y que no tuviese ninguna secuela espiritual ni nada parecido, y trabajar a partir de ahí. Con esta nueva

información, dimos vuelta atrás e iteramos para desarrollar nuestro nuevo proyecto: *Jack'n Draw*.

La propuesta definitiva: *Jack'n Draw*

La idea surgió por dos razones: en primer lugar, se nos presentó el proyecto *Burning Fury* un *fangame* realizado por otro grupo de Aula Arcade inspirado por la franquicia de videojuegos arcade *beat'em up Double Dragon*, que aprovechó el hecho de que se había publicado toda la documentación respecto al diseño del juego, y de que gran parte de los *sprites* del juego estaban en una base de datos, para copiar gran parte de los diseños por si no tenían tiempo suficiente para crearlos. En segundo lugar, pusimos en práctica el consejo de Mauricio García: CEO de *The Game Kitchen* y tutor nuestro, que nos recomendó utilizar como referencia juegos *arcades* que no se hubiesen adaptado, pues estos eran fáciles de producir y contaban con un público que le daría un gran valor al juego. Utilizando esta información y con una breve investigación se hizo el descubrimiento de que el videojuego *Snow Bros* únicamente contaba con algunas remasterizaciones que estaban destinadas a los dispositivos móviles y navegadores *Flash*; en redes sociales como *YouTube*, videos recientes sobre este juego contaban con una buena cantidad de reproducciones, lo que evidenciaba que existía un público. Para darle nuestra propia identidad decidimos un enfoque que permitiese reconocer el referente inmediatamente, pero que se sintiese nuevo. Si en nuestro referente el jugador lanza bolas de nieve a criaturas de circo para convertirlas en bolas gigantes de nivel para lanzarlas, en nuestro proyecto encarnamos a un dibujante que lucha contra sus antiguas creaciones lanzándoles bolas de papel para convertirlas a su vez en grandes bolas de papel que desechará.

Pocos días después se presentó la idea a todos los integrantes del grupo y estos se mostraron de acuerdo con la propuesta. En un tiempo muy reducido se creó el nuevo *pitching* y se propusieron nuevas ideas que reforzarían la propuesta, como una cadena de producción de niveles y la presentación de varios *concept art* que ayudarían a entender la estética del juego.



Pitching del proyecto en el que se presentaron los primeros Concept Art

Una vez el *pitch interno* fue aprobado por el equipo, fue el momento de elaborar una presentación para los tutores del proyecto. En general nuestros objetivos eran los mismos, tener un producto realizable y atractivo para un público muy específico. La estructura de la presentación fue mostrar algo de la narrativa del proyecto, enseñar sus referentes, explicar las mecánicas principales del juego, especificar el *target* del juego, justificar las razones por las que trabajaríamos en este proyecto, mostrar el calendario de trabajo y finalmente establecer nuestros objetivos principales. Una vez pensada la estructura y elaborados varios recursos visuales, la presentación se hizo el 10 de enero de 2020. La respuesta fue positiva tanto para los integrantes del equipo como para el profesorado y finalmente la propuesta fue aprobada al ser mucho más estable y realista que el anterior proyecto. Una vez todo expuesto y aprobado, el equipo entró en la *Alpha Dev* donde se desarrollarían los sistemas principales del videojuego, el GDD donde se encontraría toda la información del juego además de la identidad artística del proyecto.

4-Alpha Dev

Una vez presentado el *pitching*, llegó el momento de entrar en *Alpha Dev*, fase de la producción de un videojuego en la que se crea el núcleo de la jugabilidad y todos los sistemas principales. Al final de esta fase deberíamos tener un *PMV* (Producto Mínimo Viable), una versión jugable del proyecto en la que se pueda valorar si el proyecto es divertido e iterar en aquellos sistemas que no lo sean o sean problemáticos. Esta fase es fundamental, pues con un mal *core* el juego nunca funcionará. Por otra parte, se empezaría a trabajar en la identidad visual del juego, así que se elaboraron artes conceptuales, la estética del proyecto, los primeros *sprites* del protagonista y sus respectivas animaciones. Nuestra fecha límite era el 20 de marzo, por lo que teníamos un tiempo reducido de 2 meses para establecer todos estos sistemas.

Herramientas utilizadas

Antes de entrar en profundidad en el proyecto, era importante delimitar cuáles iban a ser nuestras herramientas durante estos meses. A continuación se relaciona una selección de herramientas de uso libre o muy económicas que se utilizaron para la producción, arte, diseño y programación de *Jack'n Draw*.

Programación y diseño

Unity 2D

El motor gráfico por excelencia de la industria *indie*. Cuenta con la posibilidad de trabajar en videojuegos en dos o tres dimensiones, así como con una gran comunidad de usuarios, por lo que si teníamos algún problema técnico no era muy difícil encontrar a alguien que ofreciera una solución; y por último, este software es gratuito a menos que la empresa que lo adquiera

ingrese más de 200.000\$ anuales, por lo que en un proyecto sin ánimo de lucro como el nuestro este era un *software* ideal.

Unity Collaborate

Extensión de *Unity* que funciona como un “controlador de versiones”. Con él, varias personas pueden trabajar conjuntamente dentro de un proyecto de *Unity*, además de ir publicando nuevas versiones actualizadas, lo que nos permite tener siempre a mano nuevas versiones del juego y, si existía algún problema fatal en el proyecto *Unity Collaborate*, acceder a versiones anteriores del proyecto, lo que nos daba la opción de retroceder en cualquier momento. Este programa es gratuito a menos que trabajen en la licencia más de dos personas a la vez, pero teniendo en cuenta el equipo con el que contábamos no necesitábamos más.

Marvel

Herramienta online que permite diseñar interfaces, muy útil en el desarrollo de videojuegos pues permite utilizar pantallas en cualquier tipo de formatos, facilitando el diseño de estas.

Arte

Aseprite

Este editor de gráficos rasterizados es una elección fantástica si se decide crear “*pixel art*” pues está diseñado tanto para crear arte estático como para realizar animaciones en este estilo. Esta herramienta fue elegida por ser la más cómoda para el juego, además de por ser una de las opciones más económicas del mercado, con un coste de solo 20 dólares.

Producción y gestión de proyectos

Hack'n Plan

La herramienta principal de la producción del videojuego. Con esta herramienta gratuita, se puede crear directamente el GDD (*Game Design Document*) de una forma muy cómoda y atractiva para los demás usuarios, pues permite crear una base de datos con el diseño de una *Wiki* que puede ser consultada con mucha facilidad y rapidez. Además, también integra una tabla *Scrum* que utilizaríamos para asignar las tareas a cada integrante del grupo, con la posibilidad de vincular esas tareas a los elementos del GDD y de que dicho miembro pudiese acceder a la información sin ninguna dificultad, lo que ahorra tener que explicar los detalles del elemento a cada persona. El único problema que presenta es que para añadir información puede ser un poco obtusa frente a otros softwares de pago como *Notion*, y también algo compleja para personas que nunca la hayan utilizado, así que es recomendable para proyectos de larga duración.

Miro

El segundo programa utilizado en la gestión del juego. Junto con *Hack'n Plan* estas herramientas fueron utilizadas para trabajar con la metodología *Scrum*. En este software gratuito, se puede crear una tabla en la que podemos crear pólisis que fueron utilizados para elaborar un cronograma, con el que se establecería todo lo necesario para el desarrollo en cada aspecto del juego, con objeto de tener una visión global de todo el proyecto.

Repositorio de arte y documentos

Google Drive

Repositorio en el que todos los integrantes del grupo importan y exportan toda la información relacionada fundamentalmente con el arte del juego, para que los programadores lo tengan a mano y lo integren en el proyecto. Este software no es muy recomendable frente a otros repositorios, pues Google Drive tiende a comprimir los archivos de forma y eso conlleva una pérdida de calidad; además, si se publican archivos de código existe la posibilidad de que

Drive los modifique, y de que al descargarlos e integrarlos en *Unity* estos den error. Pero al tener *Unity Collab* como repositorio de código, y debido a que nuestro juego era *pixel art*, dicha pérdida de calidad no era problemática.

Comunicación

Discord

Este sistema de mensajería online es ideal para la comunicación de un equipo. Es gratuito, cuenta con la capacidad de crear servidores con múltiples canales escritos o hablados y permite realizar videollamadas y compartir pantalla. También es útil en *marketing*, pues permite crear un servidor con todos los interesados del proyecto permitiéndonos tener una comunidad y una comunicación directa con ellos.

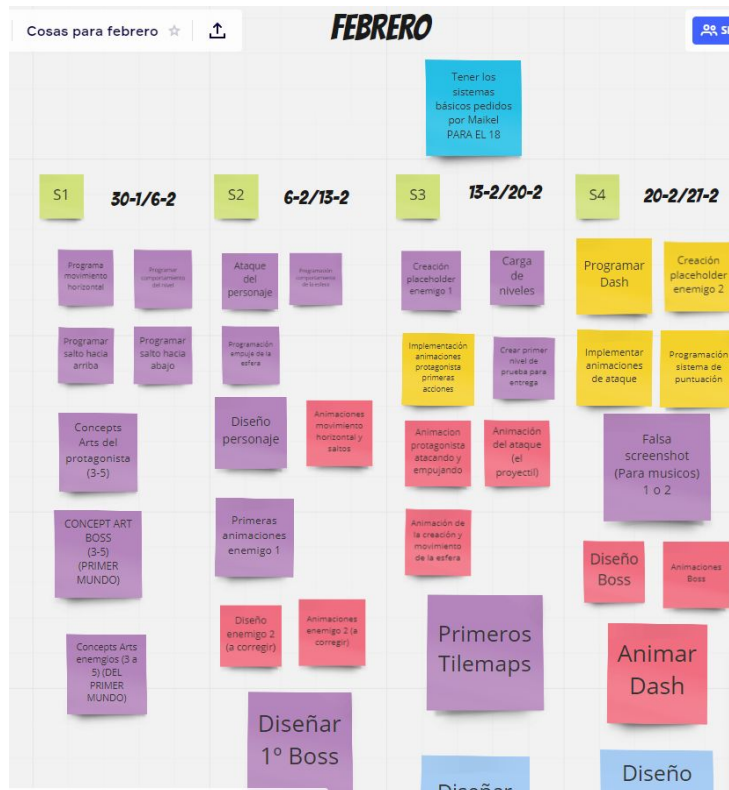
Elaboración del prototipo

Una vez establecidas las herramientas y la metodología de trabajo, era momento de trabajar en el Producto Mínimo Viable, una versión jugable de la alpha en la que se evaluará si es divertido y qué cosas se pueden mejorar o eliminar. Para ello, se establecieron dos departamentos de trabajo, diseño y programación, en los que el diseñador establecería los primeros sistemas del juego redactándolos en el GDD para que después los programadores empezasen a trabajar en estas mecánicas. Por otro lado, el departamento de arte empezaría a crear los primeros artes conceptuales, además de pensar en la identidad visual del videojuego.

Dinámica de trabajo: Elaboración de cronograma y metodología *Scrum*

Con las herramientas escogidas, era necesario establecer “cómo” y “cuándo” trabajaríamos. Para ello había que preguntarse: ¿Qué es necesario para desarrollar *Jack'n Draw*? Con ese fin, realizamos un cronograma con *Miró*. En él se establecieron todas las necesidades del proyecto para posteriormente posicionarlas en periodos de tiempo de dos semanas denominados *sprints*. Los pósts eran de diferentes colores para denominar de forma directa y muy visual la prioridad o el tipo de la tarea . Por ejemplo si eran amarillos significaba que eran prioritarios y si eran rojos, significaba que se trataba de un *bug*.

Aquí se cometieron errores derivados de no comprender totalmente la utilidad del cronograma. Esto hizo que el primer cronograma fuese desechado al poco tiempo.

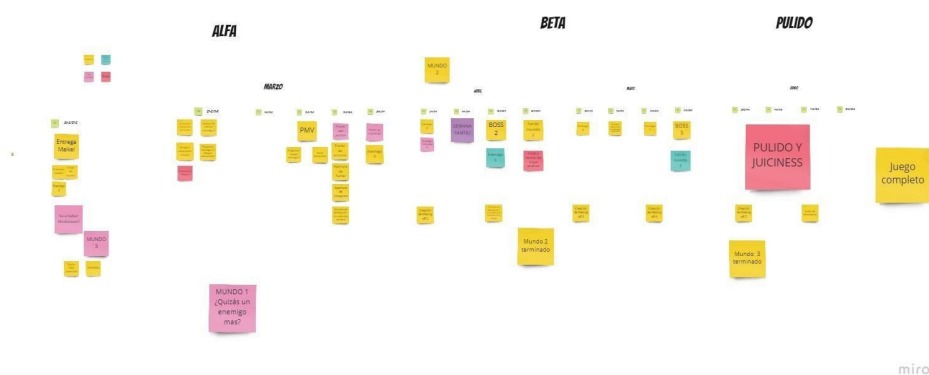


Esta fue la primera versión de nuestro cronograma, y sirve como ejemplo de lo que no es un cronograma. Esto nos da una visión del avance del proyecto y la información de los pólits debe ser lo más general posible. En las reuniones después de cada *sprint*, denominadas *sprint meetings*, se desarrollarán esas necesidades. Concretarlas al principio no tiene sentido pues el desarrollo de un videojuego no es lineal. En múltiples ocasiones se iterará y se retrocederá para pulir el proyecto. Además, es tarea de cada departamento desmigrar dichas tareas y no del productor, error que se cometía también.

Una vez elaborado el cronograma y establecidas todas las necesidades, empezamos a trabajar. Para trabajar utilizamos la metodología *Scrum*. Este proceso de trabajo es ideal para la elaboración de cualquier proyecto digital, pues permitía el desarrollo del proyecto a través de pequeños incrementos en periodos de dos semanas, que permitían revisar en múltiples ocasiones el avance de este e iterar cuando era necesario.

La dinámica *Scrum* en un videojuego funciona de la siguiente manera: en primer lugar se observa el incremento del sprint, anotando todo aquello que sea mejorable o posibles ideas que puedan surgir. Es importante que si el incremento ya es jugable, se pruebe y se pasen por todas las *features* que se hayan implementado para poder evaluarlas. Una vez jugado el incremento, es momento de que el productor indique que es necesario para el siguiente *sprint*. Para ello hay que preguntarse qué características se implementarán, qué incertidumbres se disiparán, qué contenido se añadirá y qué problemas del incremento se resolverán (García, 2020). Es importante que todos los integrantes del equipo se comprometan a decidir qué se hará de cara a la próxima presentación del incremento. Tras decidirlo, es momento de atomizar todas esas necesidades que nos hemos comprometido a hacer para posteriormente ordenarlas según su prioridad y finalmente repartir el trabajo entre todas las personas de cada departamento. Por último, todo esto se pasará a limpio en *Hack'n Plan*, asignando las ocupaciones a cada integrante del grupo.

Otro error cometido fue reducir el periodo de tiempo de los *sprints*, que pasó de dos semanas a una. Esto supuso muchos problemas para algunos integrantes del equipo, pues tenían compromisos que les impedían acudir cada semana a la reunión, lo que obligó a modificar en múltiples ocasiones el día de la reunión. Por ello, es recomendable que se respete dicho periodo de dos semanas en un proyecto de larga duración y que, si es necesario, se hagan reuniones semanales para ciertos departamentos si han surgido dudas o problemas.



Cronograma realizado para la *Alpha Dev*

Elaboración del GDD

Pero primero, ¿qué es un GDD? Son las siglas de *Game Design Document*. Es un documento que recopila todos los sistemas y contenidos del proyecto. Este documento es modificable y se va rellenando a lo largo del desarrollo. Con él, todos los integrantes del equipo saben en qué consiste el juego, cuánto trabajo hay y qué se necesita. Este documento es esencial en el desarrollo de cualquier videojuego y en el apartado de producción es necesario para estudiar la viabilidad del proyecto y para realizar una planificación mediante un cronograma. Hay que tener algo en cuenta en la elaboración de un GDD, y es que refleja lo que tiene el juego, no lo que debe tener. Es verdad que en un principio hay que documentar ciertos sistemas, pero en la práctica puede que parte de la documentación esté mal por diversas razones: quizás el personaje se mueve con demasiada lentitud, o ciertos enemigos tienen valores demasiado altos, por lo que se tendrá que iterar, cambiar dichos valores y después actualizarlos en el GDD para que posteriormente en QA se puedan verificar que los valores del juego se corresponden con los del GDD.

The screenshot displays the Hack'n Plan interface. On the left, a 'Game Design Model' tree is visible with categories like 'Character: JUGADOR (2)', 'Cutscene: Datos Del Personaje (2)', 'Mechanic: Mecanicas (8)', 'World: ESCENARIO (1)', 'Character: ENEMIGOS (6)', and 'Character: JEFES (3)'. The 'Mechanic: Saltar' item is highlighted. On the right, the 'Design Element' view for '#49 | Saltar' is shown. It includes fields for 'Element type' (Mechanic), 'Start Date', and 'Due Date'. The 'Description' section contains the following text:

CONTROLES:

- Teclas para saltar y bajar de plataforma respectivamente

Cada jugador utilizara la siguiente opción de controles por predeterminado

- J1: W y S
- J2: flecha superior y flecha inferior

Opción de mandos: joystick izquierdo en las direcciones arriba y abajo

IN GAME

Saltará a la parte superior(2 casilla), si se está pulsando en alguna dirección no avanzara mas de 1 casilla horizontal.

****ANOTACIÓN:** si se salta mientras realizas el Dash, dará un salto largo en la dirección que se esté realizando, permitiendo saltar 3 espacios horizontales.

ARTE

- sprites salto (ahora mismo el salto sera el segundo frame de la animacion de ataque)
- sprites salto hacia abajo

AUDIO

- salto

At the bottom, it shows the ID #49, creator (@joseant), creation date (Feb 6, 2020), and last update date (Mar 29, 2020).

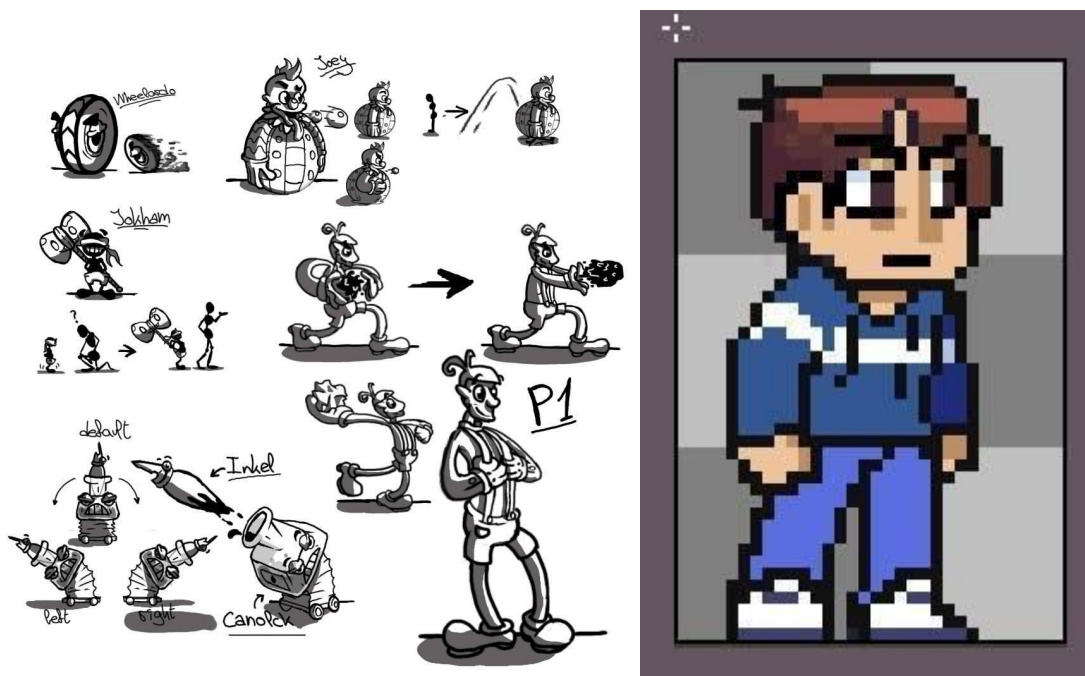
Interfaz de *Hack'n Plan*

En resumen, *Jack'n Draw* contenía los sistemas que se observan en la tabla. En él se recopilan todas las acciones que va a realizar el jugador durante la partida y en cada una se describe el sistema, cuáles son sus controles, sus parámetros, las interacciones con otros sistemas y si se trata de alguna acción del personaje, qué animación y efecto de sonido necesitan. La documentación se realizó con *Hack'n Plan* pues con este modelo es muy fácil consultar cada elemento en vez de tener todo el GDD en un documento, donde la búsqueda de ciertos elementos muy concretos pueda ser obtusa e ineficiente. De todas formas, es recomendable que en una producción todo el equipo elija una forma de elaborar el GDD, pues al final lo importante es que este sea lo más cómodo y accesible para todos los integrantes del equipo. Un GDD que no sea claro ni cómodo para todos los integrantes del equipo no tiene ningún valor.

Primeros artes conceptuales

Teníamos pensado el concepto de *Jack'n Draw*, queríamos algo que tuviese la extravagancia de *Snow Bros* pero que incluyese referencias a la cultura popular. Teníamos el concepto de Jack, un artista mediocre que se enfrenta a sus antiguos dibujos, pero no teníamos mucha idea de cómo iban a ser dichos dibujos. En un principio teníamos pensado que cada sección del juego tendría una temática propia: una *cartoon*, otra *cómic* y por último, una temática *anime* pero al final, la gran carga de trabajo hizo inviable esta propuesta. En cambio, la que sí resultó ser muy bien recibida fue la de darle una estética de tinta al juego. El personaje estaría manchado con tinta, los enemigos se componen de ella y todo el escenario estaría manchado. Por otra parte, si el juego iba a ser *Pixelart* había que pensar qué proporciones y qué estilo de pixelado usábamos. Como referentes visuales tuvimos a tres videojuegos principalmente, también con una temática de cómic: *Comix Zone*, *Scott Pilgrim Against the World* y *River City Girls*. Cada uno tenía resoluciones diferentes y, lo más importante, tipos de píxeles diferentes. A más píxeles hay más posibilidad de detalle, aunque la dificultad y el tiempo dedicado a cada animación aumenta considerablemente. Al final tomamos el estilo de *Scott Pilgrim Against the World* pues mantenía una esencia *retro* aunque contaba con los delimitados negros que transmitían esa sensación de cómic. Con esto, se empezó a trabajar en los primeros *concept art* del personaje y de los enemigos.

Respecto a las proporciones del juego, este se ve en 4:3 con barras negras a izquierda y derecha, pues nos ahorraba mucho trabajo, además de ofrecer ese toque *retro*. Además, había pensada una sorpresa para el jugador en la que en el enfrentamiento contra los jefes finales el escenario aumentaría de tamaño, pasando a un formato de 16:9. Esta idea, por falta de tiempo y recursos fue desechada: se propuso un 4:3, al que se añadirían bandas negras a ambos lados.



A la izquierda, artes conceptuales de diferentes enemigos, además de la propuesta de un posible protagonista. Al final, optamos por algo mucho más mundano.



A la izquierda, versión de 64x64 píxeles, a la derecha versión de 32x32 píxeles

Durante la realización de las primeras pruebas y diseños, se descubrió que la proporción 64x64 iba a suponer una cantidad desmesurada de trabajo para los artistas, por lo que decidimos reducir el tamaño. En primer lugar hicimos una versión de 16x16 píxeles, pero esto supondría una gran carencia de detalles y una excesiva simplicidad, por lo que al final nos decantamos por una versión de 32x32. Es muy recomendable que, si se realiza un proyecto en *pixel art*, se decida lo antes posible las proporciones del pixelado, pues elegir mal conlleva una pérdida de trabajo y tiempo bastante considerable; es recomendable consultar con otros artistas o hacer pruebas antes de la decisión final.

Programación de las primeras mecánicas

Con parte del GDD escrito, se empezó a trabajar en el código base del proyecto. De entrada se pensó en trabajar solamente con las físicas de *Unity*. Esta idea fue rechazada en un principio, pues el comportamiento de la bola no tenía nada que ver con *Snow Bros*. En cambio, en diseño se apostó por basar el movimiento en un sistema propio. En pocas palabras, el escenario estaba dividido en casillas y todo el movimiento giraba en torno a estas casillas: Jack, los enemigos y la bola se movían a x casillas por segundo y acciones como el *dash* o el salto permitían el movimiento por x casillas. Todo esto se traducían en crear unas físicas propias, lo que representó el mayor problema que tuvimos respecto a programación. El juego se sentía lento, ortopédico y tenía una gran cantidad de *bugs* que lo hacían injugable. La fecha de entrega para el PMV se acercaba, así que durante las semanas que quedaban, se decidió corregir parte de los fallos para ver si se podía arreglar lo que se había hecho. Al final, estos esfuerzos fueron en vano, pues aunque parte de los fallos se corrigieron, aparecieron muchos otros haciendo que el juego tuviese el mismo estado. Al final, el PMV que se entregó no era divertido y no tenía ningún valor. Esto supuso un gran golpe para el equipo, pues además de haber perdido mucho tiempo, tendríamos que iterar gran parte del proyecto. Si todo esto no fuese suficiente, a las pocas semanas llegó el confinamiento provocado por el Covid-19, lo que complicó significativamente los procesos.

Problemas humanos, reiteración de código y llegada de la pandemia

Tras la entrega del PMV, múltiples problemas aparecieron, haciendo peligrar en gran medida el desarrollo del proyecto. En primer lugar, el juego contaba con múltiples problemas en su núcleo, era lento, ortopédico y estaba plagado de *bugs*, aun así este problema es el más común en la industria de los videojuegos. Todos los proyectos tienen que iterar varias veces hasta obtener el producto deseado.

El mayor problema que tenía *Jack'n Draw* era humano. Ya desde el principio del desarrollo contábamos únicamente con un diseñador, pues el otro dejó el curso a las pocas semanas de iniciar el proyecto, parte de los artistas apenas podían trabajar en este, debido a que por motivos personales no podían dedicarle tanto tiempo, y uno de los programadores también se retiró del proyecto, por lo que el equipo pasó a contar con cuatro miembros frente a los ocho con que empezó. Todos los demás miembros del equipo estábamos desesperados y con la moral baja, pues al final somos seres humanos. Algo que se debe tener en cuenta es que es necesario conocer a las personas con las que trabajas y tratar con ellas lo mejor posible, sobre todo si el proyecto no les otorga ningún ingreso económico.

Como productor, este descontento era responsabilidad mía. Hablando con los integrantes del equipo y con el profesorado, saqué los siguientes fallos que debían ser solventados:

- En primer lugar, la dinámica de las reuniones. Estas estaban mal planteadas, pues no nos reuníamos todos los integrantes del equipo. En vez de ello, los jueves por la mañana se reunía arte, diseño y producción, y por la tarde diseño, programación y producción. Esto hizo que, la comunicación entre arte y programación nunca fuese directa, lo que provocó que varios *sprites* y animaciones tuviesen problemas respecto a sus proporciones y a las *hitboxes*.
- Por otro lado, la dinámica en las reuniones no funcionaba, porque hizo que varios integrantes no se sintieran atraídos en el proyecto. Nunca se veía el incremento jugable a menos que se insistiese mucho en él, y eso hizo que no se tuviese mucha información sobre el estado del juego.

Con estos errores en mente, y con recomendaciones de mi tutor, recomiendo que los *sprint meetings* sean reuniones en las que estén todos los integrantes del grupo. Con estas reuniones todos tendrán una visión clara del estado del juego, además de que conocerán mejor la razón de sus tareas, lo que les facilitará mucho el trabajo y aumentará su productividad. Por otro lado, esto les integrará en el proyecto, les dará una razón para seguir trabajando en él y no desanimarse.

Tras estudiar la razón de estos problemas, decidimos descansar durante la Semana Santa y, en la siguiente semana nos reunimos para solucionar estos problemas y llevar el juego adelante.

Soluciones para la Alpha Dev

Lo primero que había que hacer era cambiar las dinámicas del grupo. Todo el equipo tuvo una reunión y al final se decidió que la nueva fecha para los *sprint meetings* sería los domingos por la tarde. Era una fecha en la que todos estábamos libres, así que era obligatorio asistir. Además, la dinámica de grupo sería la siguiente: en primer lugar, todos los miembros del equipo mostrarían lo que habían hecho esta semana. La razón de esto fue que todos los miembros del equipo viesan qué habían hecho los demás, además de presionar para que todas las semanas mostraran sus avances, por pequeños que fuesen. Todo lo demás funciona como un *sprint meeting* común, pero con este cambio conseguimos integrar un poco más a todos los componentes del equipo. Además, los de arte trabajaron mejor pues el programador podía darle indicaciones sobre cómo debían estar las animaciones para que se integrasen correctamente. En algunas ocasiones, sobre todo en el mes de junio, algunos componentes del grupo fallaron, pero se hizo todo lo posible para mejorar la comunicación.

El segundo problema a resolver era el estado del código. Se vieron varias formas de orientar el código del juego y al final se hizo lo siguiente: tanto Jack como los enemigos funcionan respecto a las físicas de Unity, mientras que la bola funciona de la forma tradicional. Este cambio fue acertado, pues mejoró en gran medida el núcleo del juego y sorprendentemente la bola se comportaba correctamente a excepción de algunos fallos que se corregirían durante *QA*. Eso sí, reorientar todo el proyecto llevaría varias semanas, por lo que perderíamos

mucho tiempo si los demás no trabajaban en nada, además de que deberíamos habernos encontrado en *Beta Dev*. Por ello, se modificó el cronograma de forma que se empezaron a crear gran parte de las animaciones, tanto de Jack como de todos los enemigos que se iban a crear, para que cuando el programador acabase de establecer todos los sistemas del juego, tuviese material de sobra para empezar a trabajar en los siguientes enemigos y así empezar a tener incrementos jugables.

A mediados de abril, el juego tenía otro aspecto. Los sistemas eran divertidos, el comportamiento de la bola funcionaba considerablemente y ya se habían creado varios enemigos tanto en diseño como en arte para que el programador entrase directamente en la creación de contenidos.

5 Beta Dev

La gran mayoría de las incertidumbres del proyecto habían sido solucionadas, así que ahora era necesario trabajar en los contenidos del juego: enemigos, escenarios, jefes finales, personajes, *power ups*... Nuestra fecha límite para crear la mayor cantidad de contenidos posibles e integrarlos en el juego era el 31 de mayo. Esta fecha fue establecida para dejar al menos un margen de tres semanas para pulir detalles del juego y corregir *bugs*.

Para aumentar nuestra productividad, seguimos una *Vertical Slice*, metodología que en pocas palabras funciona como una cadena de montaje. A continuación se explicará cómo se creaban los enemigos de *Jack'n Draw*.

- En primer lugar se diseñan los enemigos en el GDD. En ellos se establece todas sus características y su comportamiento. Esto se presenta a los artistas, programadores y productores para dar opiniones e ideas para el diseño.
- Después era momento de elaborar artes conceptuales, o directamente elaborar un diseño en *pixel art* del enemigo. Esto segundo es desaconsejable, pues los artes conceptuales ayudan mucho a tener referencias directas cuando toque animar a los enemigos. Después de hacer los artes conceptuales, estos se presentan, se elige uno y por último se hace una lista con todas aquellas animaciones que habrá que crear.
- Una vez creadas las referencias, se empieza a animar a los enemigos. La herramienta que se utiliza es *Aseprite* y por lo general se solían hacer tres o cuatro animaciones correspondientes al *Idle*¹ movimiento, salto y ataque.
- En paralelo con el arte, el departamento de programación empieza a trabajar en los patrones de comportamiento de los enemigos utilizando *placeholders* mientras tanto. El tiempo de producción de los primeros enemigos es más largo que los demás debido a que gran parte del código, como el relacionado con el comportamiento del movimiento o las interacciones con el jugador, puede ser reutilizado sin ninguna dificultad.

¹ Se denomina *Idle* a la animación cuando el personaje se encuentra parado. Tener al personaje parado puede resultar algo brusco para el jugador, por lo que siempre se suele contar con una pequeña animación.

- Por último, se integran todas las animaciones, se testea el enemigo en un campo de pruebas y, si no cuenta con ningún problema de gran envergadura, se guarda como un objeto prefabricado para que el diseñador pueda integrarlo en los niveles del juego.

Es muy recomendable en *Beta Dev* tener un plan de creación de contenidos, pues afectará directamente a la velocidad de producción. En el desarrollo de videojuegos el tiempo es fundamental, sobre todo en esta fase, pues existe una relación proporcional entre dicha velocidad y la cantidad de contenidos que podremos crear. Debemos evitar todos los retrasos posibles para poder tener a tiempo todo lo que tenemos planeado.



Concept Art de enemigo 1



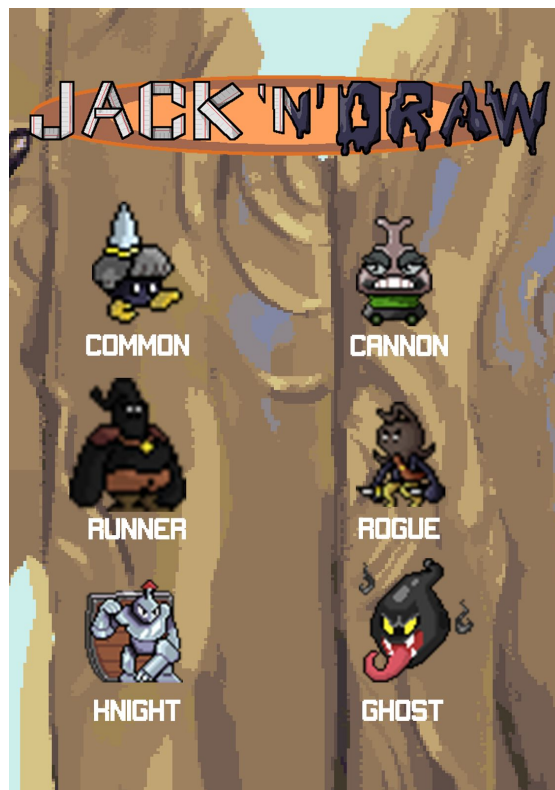
Animación de caminar de enemigo 1

Antes de ponernos a crear contenidos es necesario preguntarse: ¿Cuánto contenido vamos a crear? Es muy importante tenerlo en cuenta, pues ponernos a crear contenidos sin ningún tipo de planificación puede suponer dejar de lado tareas que pueden ser fundamentales para el buen funcionamiento del juego. En un principio, nuestro equipo tenía en mente la creación de seis enemigos diferentes, la implementación de un cooperativo local, tres jefes finales y doce power ups. Las expectativas eran altas y una falta de visión de futuro provocada por la escasa experiencia hizo que no pudiésemos crear ciertos contenidos que habrían ayudado a darle una mayor consistencia al juego, como la creación de jefes finales

Enemigos

Todos ellos pudieron realizarse gracias a seguir la cadena de producción de contenidos. Estos son muy variados, tenían sus propias animaciones y enseñaban mecánicas diversas del juego. Aun así, crear seis enemigos diferentes llevó bastante trabajo, lo que nos obligó a rehusar la creación de los jefes finales. Por ello, es muy recomendable medir el tiempo que se tarda en producir un contenido y reflexionar sobre si merece la pena destinar tiempo a ese contenido o recortar para dedicarle tiempo a otras cosas.

Una posible solución para crear muchos enemigos sin mucho esfuerzo habría sido crear versiones alternas del mismo enemigo. Modificando algunas estadísticas podríamos tener muchos contenidos a bajo coste. Para tener en cuenta esta opción hay que considerar el tipo de juego que estamos haciendo, pero en el caso de un juego arcade esto habría sido posible.



Multijugador local

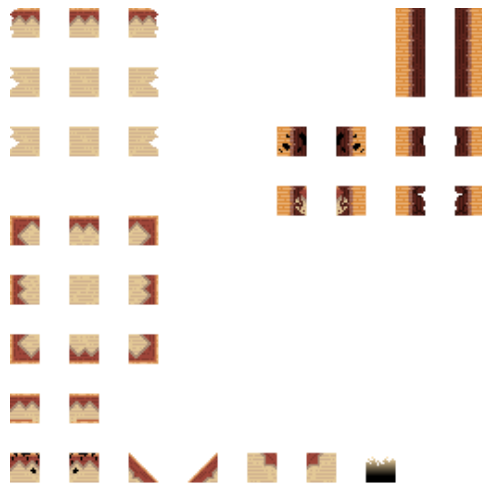
Uno de los objetivos del proyecto era que el juego constase con un multijugador local, pues al tratarse de un juego *arcade*, uno de los principales reclamos es que pueda jugarse con amigos. En términos de programación no fue nada complejo, pero a la hora de diseñar los niveles vimos que estos eran demasiado pequeños para dos jugadores, por lo que se decidió extender la anchura de los escenarios. También creamos dos aspectos más además del de Jack, para evitar cualquier confusión. Los tres aspectos eran intercambiables y muy diferentes entre sí, así que cada jugador podrá escoger un aspecto y no tener problemas a la hora de jugar. Una vez creados los tres aspectos, decidimos implementar la opción de intercambiar el aspecto en el modo de un jugador.



Aspectos intercambiables del jugador

Niveles

Trabajar con *tileset* aumentó la eficiencia en el departamento de diseño. Siendo concisos, el *tileset* es una herramienta en la que teniendo las piezas del escenario o *tiles* y los enemigos, el diseñador puede crear niveles sin apenas esfuerzo. La herramienta es muy fácil de usar, así que parte de los integrantes dieron ideas para crear una gran cantidad de niveles para el juego. Además, para cada mundo se crearon muchos niveles y, mediante el código, cada vez que el jugador jugase a *Jack'n Draw*, los niveles serían diferentes, dotando al proyecto de rejugabilidad.



Tiles utilizados. Utilizando estos elementos podemos crear escenarios muy diversos

Power Ups

Para diseñar los *power ups*, decidimos utilizarlos como recompensa para aquellos jugadores que encadenaran muertes con la bola. Se diseñaron en total 13 mejoras que daban puntuación o modificaban las estadísticas de los jugadores. Se diseñaron y animaron gran parte de ellos, aunque finalmente tuvieron que ser eliminados para acabar los *assets* en los que estábamos trabajando. Se espera añadirlos en futuras actualizaciones.



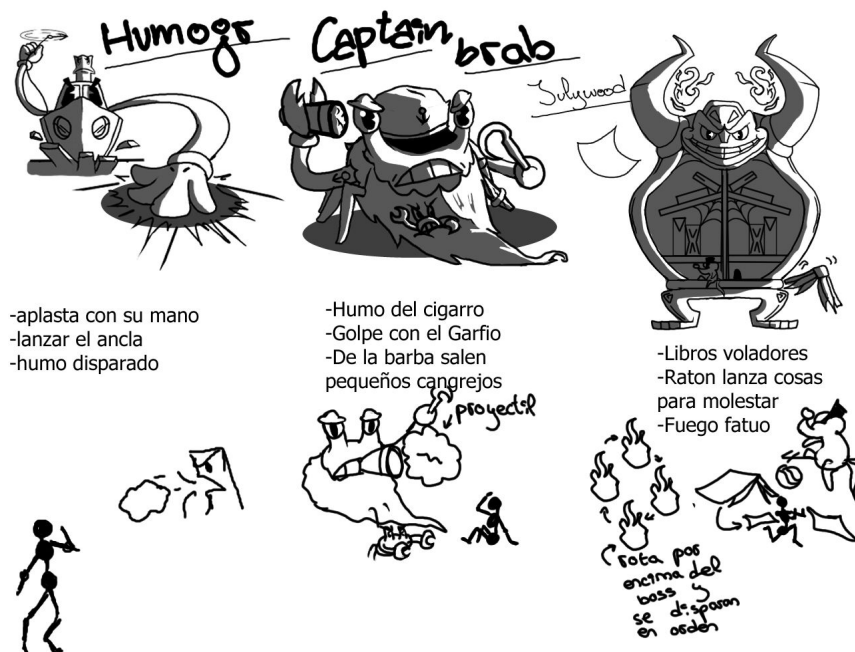
Animaciones de los Power Ups

Jefes Finales

Los Jefes de *Jack'n Draw* estaban pensados como uno de los grandes atractivos que tendría el proyecto. Tendrían otra mentalidad de diseño y supondría un gran reto para el jugador además de un espectáculo visual. Obviamente fuimos demasiado ambiciosos en esto y un mal planteamiento de *Beta Dev* nos pasó factura.

En un principio el juego iba a constar de tres mundos y al final de cada uno habría un Jefe Final. El primer jefe con el enfrentarse sería *Jullywood*, un armario gigante que tendríamos que escalar para derrotar. Tanto el diseño como el arte conceptual fueron llevados a cabo. En un principio el plan de producción nos permitía trabajar con los seis enemigos planteados y al menos un Jefe Final, pero tras la salida de un programador y de parte de los artistas apenas tuvimos tiempo para animar y programar los comportamientos de este.

De todas formas teníamos que dar un empaque final al juego, por lo que los integrantes del equipo decidimos darle al juego fases especiales en las que el jugador tendría que enfrentarse con una gran cantidad de enemigos en oleadas, en lugar de con el Jefe Final. Dependiendo de lo mucho que queramos extender el proyecto, es posible de que podamos integrar al menos este Jefe.



Artes conceptuales de los Jefes del juego. *Jellywood* fue seleccionado como el primer Jefe del juego

Valoraciones

Como comenté al principio, es muy importante definir durante el inicio de *Beta Dev* qué se va a hacer y que no. Hacer un videojuego es una carrera contra el tiempo y siempre hay que mirar cuánto tiempo nos va a llevar hacer cada *asset*. Como productor primerizo, no tuve en cuenta el tiempo que llevaría crear todos los enemigos y aunque estos pudieron ser creados, dejamos bastantes cosas por el camino que habrían mejorado el producto significativamente, como habría sido la adición de *power ups* o la implementación de un Jefe. Como recomendación para futuros proyectos, es importante tener en cuenta que es mejor tener al menos un poco de todo lo que se ha establecido que desarrollar solo un apartado. Tras llegar a finales de mes entramos en la última fase del proyecto, el *wrap-up* donde se implementará todo el material que no se haya implementado en QA y se pulirá el juego en todos sus aspectos.

6: QA, *wrap up* y publicación

Esta es la última fase del desarrollo de un videojuego antes de su lanzamiento. En ella se pulen todos los elementos esenciales que no hayan sido pulidos, se descarta todo lo no esencial que falte por pulir, se reemplaza los *placeholders* por el arte actualizado si no se hizo en *Betadev*, se arregla cualquier cabo suelto y sobre todo, se eliminan *bugs* sistemáticamente. Esta fase es de las más complicadas en el desarrollo de cualquier videojuego, debido al escaso tiempo que se tiene para salir de esta fase y por lo general puede percibirse el juego como “roto”. Esto se debe a un término llamado “deuda tecnológica”, en la que en *Alpha Dev* y *Beta Dev*, si se producen *bugs* que no sean fatales para el correcto desarrollo del videojuego se dejan para QA. Es muy importante mantener la deuda tecnológica lo más baja posible, pero al final siempre van a haber más *bugs* de lo esperado, pues es común la situación de corregir un *bug* y que detrás de él aparezcan otros tres que empeoren aún más la experiencia de juego. Además, debe añadirse el estado emocional del equipo. Todos estábamos cansados, habían pasado muchos meses de desarrollo y contábamos con diversos problemas que complicaron el desarrollo del juego, como la falta de ciertos *assets* y de tiempo por parte de algunos artistas.

Para la corrección de *bugs*, seguimos el siguiente proceso. En primer lugar, la persona que estuviera más libre testeaba el juego, mientras se iban anotando todos los *bugs* que iban produciéndose. Así lo hicimos nosotros, pero también es recomendable crear sesiones de *playtesting* con amigos u otros desarrolladores; o, si contamos con una base de seguidores, se puede publicar el juego para que sean los propios jugadores quienes vayan jugando y notifiquen en un foro todos los errores que se vayan produciendo. El siguiente paso fue priorizar todos los errores en *Hack'n Plan* para evaluar respecto al tiempo que teníamos, cuáles eran importantes y cuáles no tanto. Una vez priorizados, el equipo trabajaba en eliminar estos problemas y, finalmente, se hacía otra sesión de testeo para ver si se producían más *bugs*. Esto se realizaba en bucle hasta no detectar más fallos o haber agotado el tiempo.

A nuestro equipo se le acabó el tiempo: el 28 de junio tuvimos que entregar el juego en Aula Arcade, aunque había ciertos elementos del juego, como la música o la implementación de

ciertas animaciones, que no fueron posibles. Aun así, el juego estaba completo y bien pulido, por lo que se podía decir que estaba acabado. Lamentablemente, ciertos objetivos propuestos, como la elaboración de una campaña de *marketing* y la creación de una comunidad de al menos 50 personas, fueron demasiado ambiciosos. De todas formas, se planeó lanzar el juego oficialmente en septiembre en la plataforma *Itch.io* junto con un tráiler para tenerlo allí y poder utilizarlo como portfolio.

El juego mantenía una estética *retro* y transmitía el mismo tipo de *gameplay* que *Snow Bros*, aunque la posibilidad de crear nuevos niveles y desafíos con la implementación del *dash* lo diferenciaba de su referente. Además, utilizar un diseño de niveles de *tileset* fue un acierto, pues permitía la rejugabilidad de este y el modo desafío sustituía al Jefe con eficacia, aunque es posible que trabajemos en un Jefe Final para cuando lancemos el juego en *Itch.io* para darle un mayor empaque.

El juego puede ser jugado aquí:

<https://drive.google.com/file/d/1cJSqMc8SPztmVKhkJjenRioIrQIi9z89/view?usp=sharing>

7:Conclusiones

Desde la preproducción de *Jack'n Draw* hasta su publicación, el desarrollo de este videojuego ha sido realmente complicado y enrevesado en todos los aspectos del proyecto. En múltiples ocasiones, somos incapaces de ver el trabajo que hay detrás de un videojuego, por simple que sea. Muchos jóvenes desarrolladores hemos cometido el error de subestimar la dificultad que pueda tener desarrollar un videojuego cuando la realidad es que, si somos incautos, gran parte de esos proyectos nunca saldrán a la luz. Libros y diarios como *Blood, Sweat and Pixels* nos muestran que esta dificultad no se encuentra solo en *developers junior*, sino también en grandes estudios AAA, que han visto arruinarse muchos proyectos por ello. Este *post-mortem* es una experiencia más de la realidad del videojuego: un mundo apasionante pero a la vez aterrador, lleno de incertidumbres e inseguridad. Por ello mismo, considero que es necesario normalizar el rol del productor en el videojuego. Teniendo en cuenta todos los problemas técnicos que conlleva llevar adelante un proyecto, no contar con una organización sólida puede convertir un desarrollo en una pesadilla insostenible, sobre todo si se trata de un trabajo profesional, con graves casos de *crunch* en grandes estudios como Naughty Dog (Kotaku, 2020a) o Rockstar (Kotaku. 2020b). Es necesario tener una organización clara, tanto de objetivos como de metodología de trabajo para crear con ello un ambiente de trabajo sostenible donde el videojuego pueda salir adelante.

Personalmente, producir *Jack'n Draw* me ha otorgado perspectiva. Tenía asentada en mi cabeza que producir un videojuego era lo mismo que producir un contenido audiovisual tradicional y me equivoqué. Existen muchos factores que lo *diferencian* de los demás medios audiovisuales, y debido a mi falta de conocimientos en este se produjeron situaciones desfavorables que dificultaron el desarrollo del juego. Aun así, a lo largo del proyecto, fui recabando perspectiva y metodología, algo que me permitirá producir y llevar adelante futuros proyectos con mayor eficacia. Algunos de los valores que he aprendido tras producir este juego han sido: “Nunca pienses que hacer un tipo de juego es fácil a menos que lo hayas hecho antes”, “Ten siempre en cuenta con qué tipo de personas trabajas, conócelas y aprende cómo trabajan para aprovechar todo su potencial” y “Trabaja siempre con fechas

autoimpuestas, nunca retrases tu proyecto a menos que sea indispensable, pues si no, el juego nunca se publicará”.

Para todos aquellos que quieran aprender sobre el mundo de la producción de los videojuegos, recomiendo encarecidamente trabajar en proyectos mucho más pequeños que el mío. Participar en *game jams* y eventos similares es un punto de partida ideal, pues esto aporta una idea de cómo es realmente trabajar en un videojuego, y permite tener un espacio seguro para desarrollar tus ideas. El trabajo humilde y paciente es el cimiento necesario para los grandes proyectos del futuro.

8: Bibliografía

(VVAA. 2020. *Libro Blanco Del Desarrollo Español De Videojuegos 2019*. 1st ed. Madrid).

(García, Mauricio., 2020. *Desarrollo Videojuegos: Cómo Sacarle Provecho Al Sprint Meeting 2 - Prodevtion*. [online] CEOindie.me. Disponible en: <<https://ceoindie.me/2018/03/23/sprint-meetings-scrum-desarrollo-videojuegos-2/>> [Acceso 19 de Julio 2020])

Schreier, Jason (2017). *Blood, Sweat, and Pixels: The Triumphant, Turbulent Stories Behind How Video Games Are Made* (1st ed.). Harper.

(Kotaku. 2020a. *As Naughty Dog Crunches On The Last Of Us II, Developers Wonder How Much Longer This Approach Can Last*. [online] Disponible en: <<https://kotaku.com/as-naughty-dog-crunches-on-the-last-of-us-ii-developer-1842289962>> [Accessed 27 July 2020])

(Kotaku. 2020b. *Inside Rockstar Games' Culture Of Crunch*. [online] Disponible en: <<https://kotaku.com/inside-rockstar-games-culture-of-crunch-1829936466>> [Accessed 27 July 2020].)

Requena Molina, Nacho. (2020). Entrevista a The Game Kitchen. *Manual*, (5), 163. Retrieved 31 August 2020, from.

Pérez Rufi, José Patricio (2015). *El modelo europeo de desarrollo de videojuegos* (1st ed., pp. 170-177). Editorial Síntesis.

Wijman, Tom (2019) Newzoo Global Games Market Report 2019 , Disponible en: <https://newzoo.com/insights/trend-reports/newzoo-global-games-market-report-2019-light-version/> .)