Universitat Oberta de Catalunya

uoc.edu

Ant’s Scape: Videojuego realizado con Unity

Gabriel Gómez Martínez
Grado de ingeniería informática
Videojuegos

Guillermo García Romero
Joan Arnedo Moreno

03/01/2021


Esta obra está sujeta a una licencia de
Reconocimiento-NoComercial-SinObraDerivada
3.0 España de Creative Commons

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Ant's Scape: Videojuego realizado con Unity*
Nombre del autor:	*Gabriel Gómez Martínez*
Nombre del consultor/a:	*Guillermo García Romero*
Nombre del PRA:	*Joan Arnedo Moreno*
Fecha de entrega (mm/aaaa):	01/2021
Titulación:	*Grado de ingeniería informática*
Área del Trabajo Final:	*Videojuegos*
Idioma del trabajo:	*Castellano*
Palabras clave	Minesweeper, Mobile, Casual

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados i conclusiones del trabajo.*

El presente trabajo final del grado de ingeniería informática dentro del área de videojuegos en la UOC consiste en la creación de un videojuego con el motor gráfico Unity.

En el documento explicaremos las fases que ha atravesado el proyecto desde su idea inicial hasta la implementación final de su jugabilidad, pasando por el diseño de los personajes, modos de juego, implementación de sonidos, etc. Por otro lado, explicaremos también su parte más técnica con los scripts realizados en C# y los algoritmos mas importantes del proyecto.

En último lugar, adjuntaremos en el documento tanto los videos como el enlace del proyecto para poder comprobar el resultado final del aplicativo.

Abstract (in English, 250 words or less):

This document contains the final degree project for the degree in computer engineering within the area of videogames at the UOC university. This project consists of creating a videogame using Unity Game Engine.

In the document, we will explain the project from initial phase to the final implementation of its gameplay. We will also explain the design of the characters, game modes, implementation of sounds, etc. On the other hand, we will expound its technical part, explaining the scripts (C#) and most important algorithms of the project.

Lastly, we will attach videos and the project link, so that the final result of the application can be verified.

Índice

Lista de figuras

1. Introducción

1.1 Contexto y justificación del Trabajo

Como finalización del grado de ingeniería informática es necesario la presentación del TFG, en mi caso he elegido como tema a tratar los videojuegos. El reto propuesto es la creación de un videojuego para demostrar los conocimientos obtenidos durante este curso.

El resultado que se quiere obtener es un videojuego con la jugabilidad, personajes y niveles correctamente implementados, teniendo en cuenta el tiempo y recursos de los que disponemos.

1.2 Objetivos del Trabajo

Los objetivos presentados en el documento serán los siguientes:

- Historia y desarrollo del genero del videojuego.
- Definición del videojuego a realizar:
  - Arte
  - Jugabilidad
  - Generación de niveles
  - Definición de personajes
  - Mecánicas implementadas
  - Objetos a utilizar
  - Implementación de la cámara
  - Implementación de sonidos
  - Modos de juego
  - Implementación de menús/UI
- Elección de plataforma donde desarrollar el videojuego.
- Elección de las herramientas utilizadas para el desarrollo.
- Versión ejecutable del proyecto.
- Estudiar la viabilidad del producto.
- Establecer una ruta de mejoras para el futuro del proyecto.

1.3 Enfoque y método seguido

La estrategia escogida para llevar a cabo el trabajo es la creación de un producto nuevo, basándonos en una idea ya existente (el juego buscaminas) y algoritmos existentes (Floodfill [1]) pero añadiéndole un aspecto propio, más funcionalidades y cambiando las normas del juego original para darle otro enfoque.

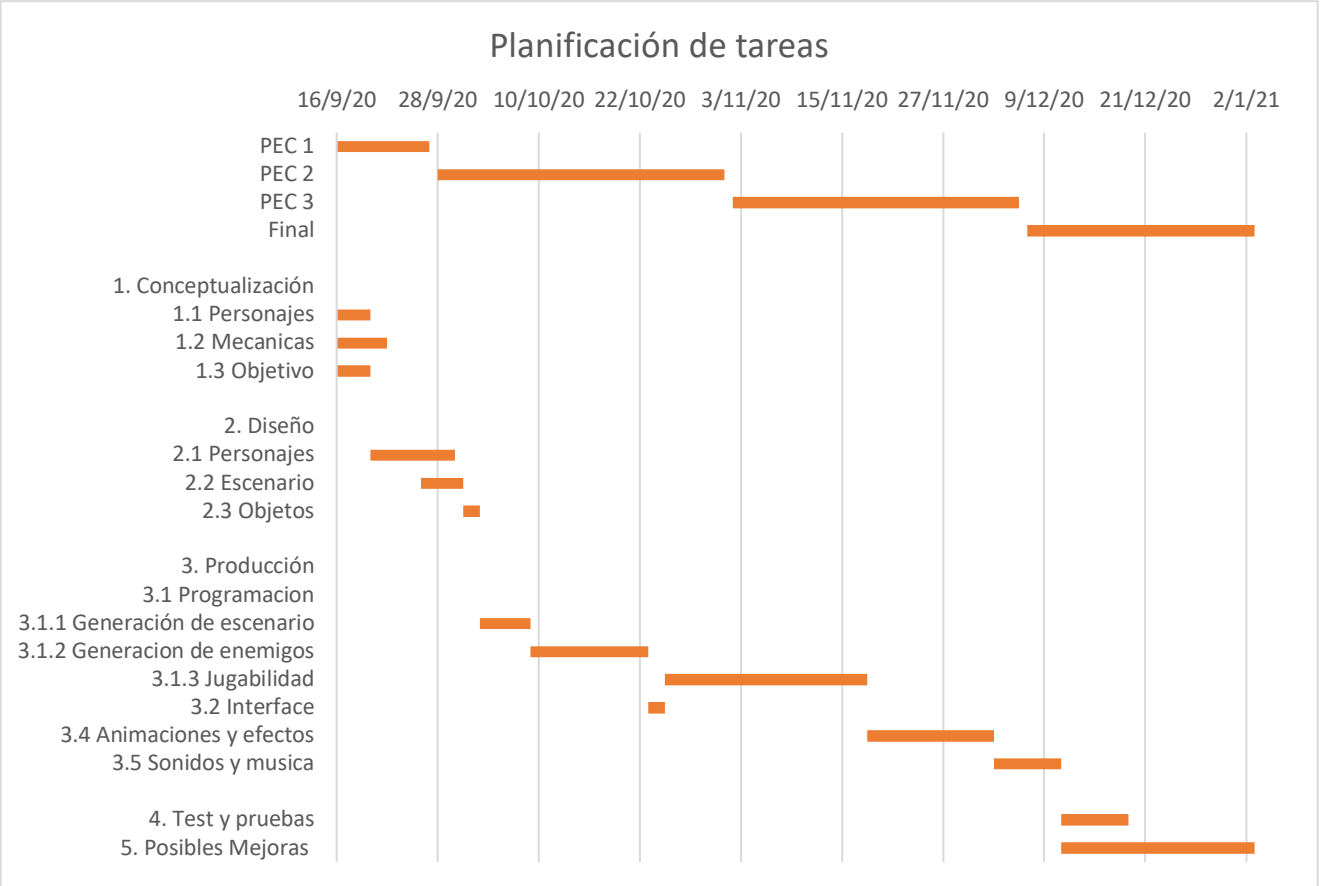
La creación de un videojuego es un proceso largo y complejo que requiere tener conocimientos en diferentes áreas, como en el ámbito de la programación, dibujo 2D, modelaje 3D, música, animaciones, etc. Por ese motivo tenemos que tener en cuenta nuestro conocimiento y, sobre todo, el tiempo que tenemos para realizarlo. Por ello, aunque usaremos assets propios como los personajes,

fondos, animaciones o niveles, también usaremos recursos gratuitos, como es el caso de la música, sonidos fx y las imágenes utilizadas para la creación de la UI.

Por otro lado, vamos a crear el juego apoyándonos en el motor de videojuegos Unity, cuyo uso es libre y nos ayudara mucho en la implementación de todos los assets comentados anteriormente. Para "dar vida" al videojuego, Unity se apoya en scripts realizados con el lenguaje de programación C#. En estos scripts implementaremos la lógica del videojuego, donde se podrá reflejar lo aprendido durante este grado.

1.4 Planificación del Trabajo

El trabajo ha sido realizado en base a la planificación de la asignatura. En esa planificación por semanas hemos dividido las tareas a realizar de la siguiente manera:



Con todo ello, el desglose de tareas quedaría de la siguiente manera:

- PEC 1
  - En esta PEC no realizaremos entrega de código, pero finalizaremos la conceptualización de los personajes, mecánica y objetivo del juego.
- PEC 2
  - Diseño de los personajes completados
  - Generación de niveles aleatorios

    - Enemigos diseñados e implementación de uno de ellos
    - Interface del juego
    - Establecer un primer punto de la jugabilidad: Movimiento de personaje, desbloqueo de celdas y perder/ganar la partida
- PEC 3
    - Jugabilidad completada con todos los enemigos
    - Todos los modos de juegos implementados
    - Sistema de puntuación
    - Animaciones de personajes
    - Implementación del sonido y la música
- Final
    - Búsqueda de sonidos y música para el juego final
    - Refactorizar código y posibles mejoras
    - Pruebas y test

El tiempo invertido en cada tarea sería el siguiente:

Tarea	Fecha de inicio	Días	Fecha final
PEC 1	16/09/20	11	27/09/20
PEC 2	28/09/20	34	01/11/20
PEC 3	02/11/20	34	06/12/20
Final	07/12/20	27	03/01/21
1. Conceptualización			
1.1 Personajes	16/09/20	4	20/09/20
1.2 Mecánicas	16/09/20	6	22/09/20
1.3 Objetivo	16/09/20	4	20/09/20
2. Diseño			
2.1 Personajes	20/09/20	10	30/09/20
2.2 Escenario	26/09/20	5	01/10/20
2.3 Objetos	01/10/20	2	03/10/20
3. Producción			
3.1 Programación			
3.1.1 Generación de escenario	03/10/20	6	09/10/20
3.1.2 Generación de enemigos	09/10/20	14	23/10/20
3.1.3 Jugabilidad	25/10/20	24	18/11/20
3.2 Interface	23/10/20	2	25/10/20
3.4 Animaciones y efectos	18/11/20	15	03/12/20
3.5 Sonidos y música	03/12/20	8	11/12/20
4. Test y pruebas	11/12/20	8	19/12/20
5. Posibles Mejoras	11/12/20	23	03/01/21

1.5 Breve sumario de productos obtenidos

El producto obtenido en la finalización del proyecto será una fase beta del videojuego que estará compuesto por números scripts y assets propios que se desglosaran en apartados posteriores.

1.6 Breve descripción de los otros capítulos de la memoria

Los capítulos que nos encontraremos en el documento son los siguientes:

- Capítulo 1: Introducción

  Se trata el contexto y justificación del trabajo, así como los objetivos, metodología y planificación del proyecto. Este punto es una primera toma de contacto para afrontar el tema y el contenido del documento.

- Capítulo 2: Estado del arte

  Se comenta la evolución del genero del videojuego que trataremos, así como los motores gráficos más actuales.

- Capítulo 3: Definición del juego

  Primer capitulo de nuestro videojuego, en el cual nos centraremos en sus características principales, personajes e interacciones entre ellos. Por otro lado, remarcaremos el objetivo del juego y sus mecánicas.

- Capítulo 4: Jugabilidad

  Capítulo centrado en la parte jugable del juego, explicando detalladamente sus mecánicas, como por ejemplo el movimiento del personaje, objetos, sistemas de puntuaciones o el mecanismo para descubrir celdas.

- Capítulo 5: Diseño técnico

  Este capitulo va muy ligado al capitulo anterior, sin embargo, en esta parte nos adentraremos mas en la parte técnica, como los scripts utilizados la implementación del sonido, prefabs, etc. Por otro lado, seleccionaremos una serie de aplicaciones que nos ayudaran a realizar el proyecto, como un motor grafico o herramientas de dibujo para los sprites.

- Capítulo 6: Diseño de niveles y modos de juego

  Explicaremos los diferentes modos de juego que tiene el proyecto y la forma en la cual se generan los niveles de manera aleatoria.

- Capítulo 7: Manual de usuario

  Información e instrucciones para poder jugar, así como una ligera explicación de las diferentes UI que tenemos en el videojuego.

- Capítulo 8: Mejoras del proyecto

  Teniendo en cuenta el tiempo que tenemos para realizar el videojuego, una vez finalizado tendremos mucho margen de mejora para que llegue a ser un juego mucho más atractivo. Revisaremos diversos puntos que nos podrían ayudar a mejorar el juego.

- Capítulo 9: Viabilidad del producto

  En este capítulo se tratará la viabilidad del producto en el mercado.

- Capítulo 10: Conclusiones

  Conclusiones finales del proyecto. Haremos una trayectoria por la planificación propuesta y si se llegó a cumplir, posibles críticas del trabajo, mejoras surgidas en medio del desarrollo, etc.

- Capítulo 11: Glosario

  Definición de la terminología utilizada en el documento.

- Capítulo 12: Bibliografía

  Recursos bibliográficos utilizados para el desarrollo del documento.

2. Estado del arte

2.1 Genero del juego

Hoy en día existen multitud de géneros en el mundo del videojuego incluso tenemos ejemplos de multigénero, como es el caso del juego indie Braid (puzles y plataformas). Además, generación tras generación nace algún genero nuevo o algún juego que "rompe moldes", dando pie a una nueva base para poder crear nuevas experiencias. Sin embargo, podemos encontrar algunos juegos que tienen una base tan sólida que son igualmente disfrutables hoy en día, como por ejemplo el Tetris.

El Tetris es uno de esos juegos donde parece que no pase el tiempo, su jugabilidad sigue intacta con el paso de los años y siguen sacando nuevas versiones en las nuevas generaciones: Tetris 99 (Nintendo Switch) y Tetris effect (Xbox One y PS4):

Ilustración 1 Tetris effect y Tetris 99(https://www.tetriseffect.game/ y Nintendo.es)

El Tetris forma parte del género de videojuegos de lógica, también conocidos como videojuegos de inteligencia o videojuegos de puzle, los cuales se caracterizan por exigir agilidad mental al jugador [2]. Dentro de este género nos podemos encontrar diferentes formas de afrontar el reto, ya sean problemas de lógica, matemáticos, estrategia, etc. Es uno género complejo de describir ya que, aunque formen parte del mismo tipo de videojuego, podemos comprobar las grandes diferencias entre el Tetris, Braid o el Profesor Layton.

Detallando un poco más la historia de este género, se considera que el primer puzle en videojuegos fue el Hunt the Wumpus, programado en 1972 por Gregory Yob para Basic, el cual solo funcionaba por texto, cuyo objetivo era encontrar a Wumpus en un mapa. Este concepto fue evolucionando dando pie a versiones gráficas y el nacimiento de Zork en 1980. Sin embargo, sería la llegada del Tetris, en esa misma década (1984), la que marcaría un hito mundial en el mundo de los videojuegos [3].

El gran atractivo de este tipo de juegos es que no se sostiene sobre una base narrativa ni requiere que el jugador tenga una habilidad especial para poder jugarlo, tampoco es necesario grandes tutoriales. Realmente lo que te mueve a jugarlos es el impulso por resolver lo que se presenta como un misterio, es algo

que siempre ha estado con las personas mucho antes de la llegada de los videojuegos. Por ello, este tipo de juego se perfila como ese género que seguirá con nosotros durante un largo tiempo, y así lo ha demostrado a lo largo de los años [3].

Si seguimos con la historia de los videojuegos de puzles, podemos encontrar otro de los precursores de este género, en el cual esta fuertemente inspirado este proyecto, el buscaminas [4].

Ilustración 2 Buscaminas (wikipedia.org)

Como se puede comprobar en las imágenes estos juegos son bastante autoexplicativos y únicamente se basan en su componente lógico. Sin embargo, con el avance tecnológico y las nuevas ideas que surgían sobre este género empezaron a darle un contexto e historia a este tipo de proyectos, incluso creando puzles cada vez mas interactivos. En esta nueva vertiente del genero podemos encontrar los Lemmings, videojuego de 1991, donde tendremos que conseguir que unas criaturas consigan superar ciertos obstáculos [5].

Ilustración 3 Lemmings (ign.com)

En los siguientes años siguieron expandiendo el género de puzles, uniéndolo a otros géneros como pueden ser plataformas, por ejemplo Braid (2008) o aventuras graficas donde encontramos la saga del Profesor Layton (2007).

Ilustración 4 Braid (3djuegos.com) y Profesor Layton (guiesnintendo.com)

Esta época fue clave para el mercado del videojuego y sobretodo para el género que estamos tratando. Hasta el momento solo hemos tratado con videojuegos nacidos para videoconsolas/PC, sin embargo, en este punto se abre un nuevo camino donde el mercado es muchísimo mas amplio, hablamos del mercado móvil.

Este mercado consigue que se expandan los videojuegos a toda persona que tenga un móvil, llegando a un público que no conocía nada de este mundo. Con este gran cambio de mentalidad en 2009 nacería Angry Birds un juego de puzles donde teníamos unos personajes coloridos y carismáticos, y unas mecánicas muy sencillas dando lugar a un juego apto para todos los públicos.

Ilustración 5 Angry Birds(apps.apple.com)

Angry Birds supuso en verdadero boom en el género [6], pero si avanzamos hasta el año 2014, llegamos al que fue y sigue siendo uno de los lideres en ingresos del mercado móvil, Candy Crush. Un juego de puzles que consiste en hacer combinaciones de caramelos de tres o mas de un mismo color [7]. Aunque no es una idea propia, ya que esta inspirado en BejeWeled (2001), si que fue el que expandió el tipo de juego a un publico muchísimo más amplio.

Ilustración 6 Bejeweled (gamespot.com) y Candy Crush (pinterest.com)

Como podemos comprobar, el paso de los años parece que no afecte a este género, incluso aprovecha nuevas tecnologías de manera muy acertada como observamos en su inclusión en el juego móvil. Por otro lado, surgen nuevas ideas de cómo aplicar juegos de lógica a diferentes géneros. Por ello, la evolución de este género es difícil de predecir, pero podemos estar seguros de que los juegos de lógica estarán con nosotros siempre.

2.2 Engines y kits de desarrollo

Con el paso de los años, la evolución de los videojuegos ha ido de la mano con las mejoras de las herramientas de desarrollo. Hace 20 años era impensable que tuviéramos motores de desarrollo como por ejemplo Unity o Unreal. Por ello, conforme la tecnología avanzaba también lo hacían las herramientas con las que se hacían estos videojuegos, no solo a nivel de motores gráficos, también tenemos las herramientas de edición de sprites (Photoshop), modelado 3D (Blender), edición de sonido, etc.

Por otro lado, un punto a destacar con el avance de estas herramientas es el uso libre que tienen los motores hoy en día como por ejemplo Unity y Unreal. Esto ha sido un cambio de mentalidad enorme respecto hace 10 o 15 años. Ahora todo el mundo tiene acceso a estos motores gráficos, esto ha dado lugar a una explosión de juegos indies en toda clase de plataformas. Como hemos visto en el apartado anterior, gracias al uso de estos programas hemos podido ver juegos como Braid, The Talos Principle o The Witness.


Ilustración 7 The witness (ign.com) y The Talos Principle (Store.steampowered.com)

Por ello, podemos decir que, aunque el género de puzles, es un tipo de género con un objetivo muy básico ha sabido adaptarse a los tiempos modernos y, gracias a las nuevas herramientas, aun nos puede sorprender.

3. Definición del juego

3.1 Descripción del juego

Juego de puzles donde el jugador tendrá que ir eludiendo enemigos y rescatando compañeros para poder completar la partida. Es un juego basado en celdas tapadas donde el jugador tendrá que ir descubriendo que se encuentra detrás de cada una. En las celdas podemos encontrar pistas sobre peligros cercanos, compañeros a salvar o enemigos.

3.2 Historia, ambientación y/o trama

El proyecto es un juego de puzles donde la trama e historia no tiene mucha relevancia. Aunque si que tiene un objetivo guiado por la ambientación.

El mundo en el que se desarrolla tiene una ambientación desenfadada y colorida, donde una hormiga tendrá que rescatar a sus amigas que están debajo tierra. Para encontrar a las amigas, el personaje principal tendrá que sortear a los enemigos que estarán presentes: cigarras, topos y la lengua de los osos hormigueros.

3.3 Definición de los personajes

Como hemos avanzado en el apartado anterior, el juego dispone de diferentes personajes:

- **Hormiga principal**: La cual se encargará de buscar a sus amigas a lo largo de la aventura. Es el personaje que manejará el usuario.
- **Hormigas compañeras:** Hormigas perdidas a las que tendrá que encontrar la hormiga principal y de esta manera conseguiremos más puntos y acabaremos la partida.
- **Cigarra:** Enemigo principal del juego y al que más nos encontraremos. Ocupará un espacio de celda (como las bombas del buscaminas) y no será visible por el jugador hasta destapar la casilla.
- **Oso Hormiguero:** Enemigo visible ya que estará en la parte alta de la tierra, sin embargo, lo que no será visible es la lengua que ocupará un numero de celdas aleatorio.
- **Topo:** Enemigo que tiene más posibilidades de encontrarse en la parte mas inferior de la tierra. Este enemigo ocupara 4 casillas.

3.4 Interacción entre los actores del juego

La interacción entre los distintos actores del juego es simple, por una parte, disponemos de los enemigos (cigarras, osos hormigueros y topos). Si el jugador (hormiga principal) colisiona con algunos de estos enemigos morirá y dará paso al Game Over.

Por el contrario, si se encontrara con una hormiga compañera se sumarán puntos y si las salvamos a todas ganaremos la partida.

Por otro lado, tenemos los objetos del juego, que tendrán un apartado especial en los siguientes capítulos, ya que la interacción con las celdas cambiara dependiendo del objeto escogido.

3.5 Objetivos planteados

El objetivo principal que se le plantea al usuario es superar los niveles consiguiendo todas las hormigas necesarias para ello. Sin embargo, y derivado de este punto, una parte importante del juego es la puntuación obtenida al acabar la partida. Esto puede ayudar a que el jugador quiera intentar batir su anterior record.

En el apartado de puntuación detallaremos como conseguir este objetivo.

3.6 Concept Art

En este capitulo mostraremos un ejemplo de los diferentes recursos gráficos creados para el videojuego. No los pondremos todos, ya que las animaciones hacen que tengamos múltiples imágenes, pero si que pondremos todos los personajes/objetos en alguna de sus distintas animaciones:

- **Hormiga principal**

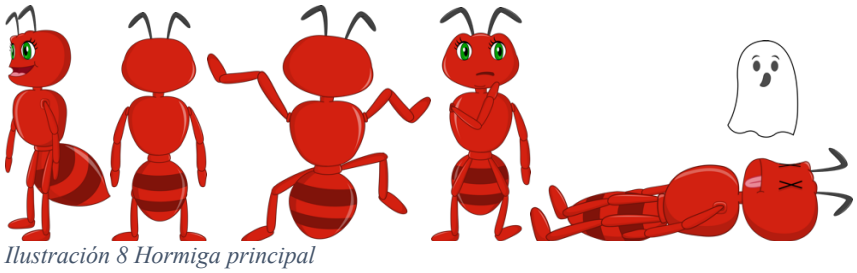


Ilustración 8 Hormiga principal

- **Hormiga amiga**

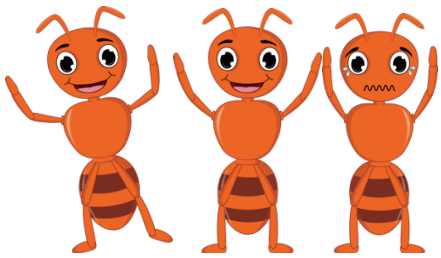


Ilustración 9 Hormiga amiga

- **Topo**

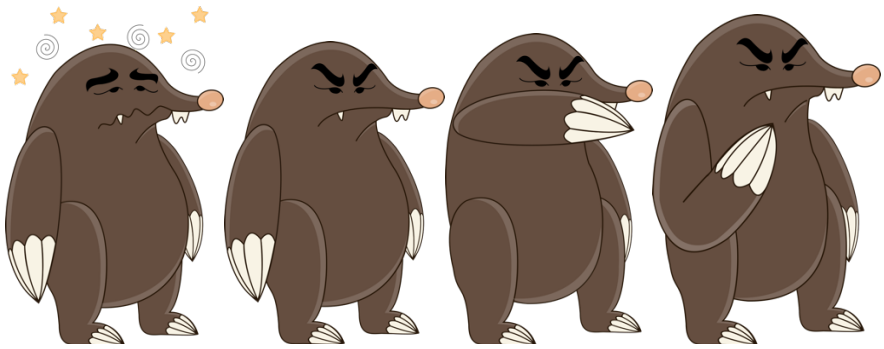


Ilustración 10 Topo

- **Cigarra**

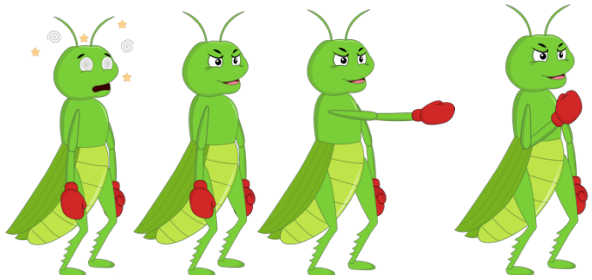


Ilustración 11 Cigarra

- **Oso Hormiguero y su lengua**

Ilustración 12 Oso hormiguero

Ilustración 13 Lengua del oso hormiguero

- **Objetos**

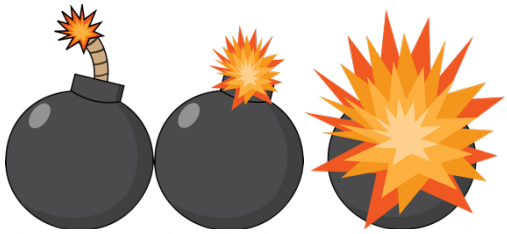


Ilustración 14 Objeto bomba

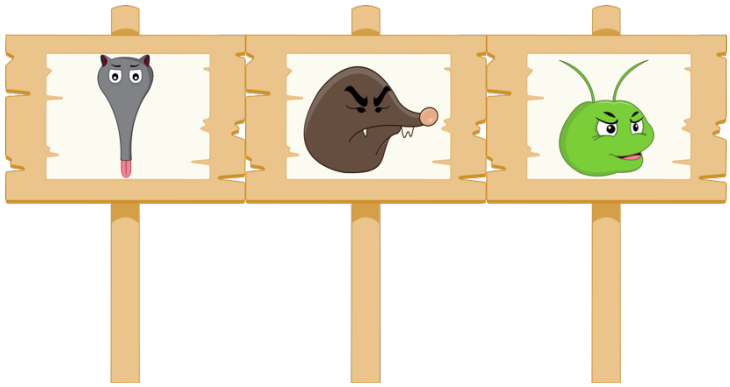


Ilustración 15 Señales de enemigos

- **Imágenes para componer los niveles**

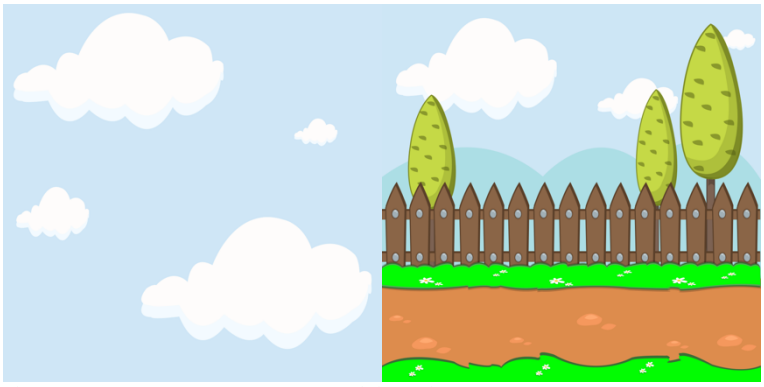


Ilustración 16 Imagenes superiores del nivel

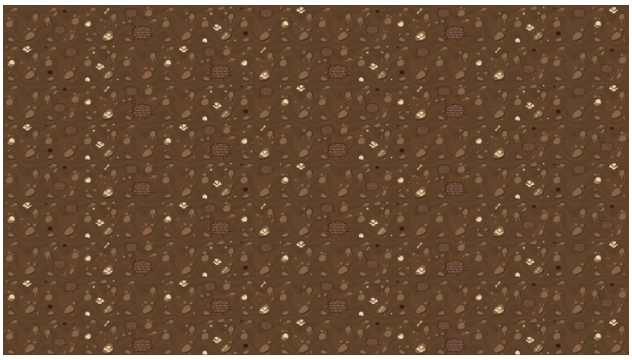


Ilustración 17 Fondo de pantalla

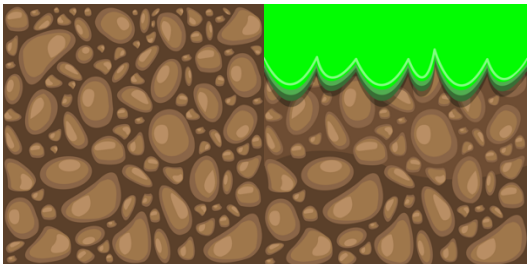


Ilustración 18 Celda tapada y celda superior

3.7 Plataforma destino

La plataforma destino es el móvil, ya que es una plataforma que no para de crecer. Por otro lado, también es un tipo de juego ideal para hacer una partida rápida y dado que es para todos los públicos abarca un gran mercado.

4. Jugabilidad

4.1 Descubrir celdas y movimiento de personaje

En el juego, podemos ir destapando celdas para descubrir que nos aguarda en esa celda. Sin embargo, y al contrario que el buscaminas, el usuario no podrá pulsar en cualquier celda, sólo podrá pulsar en las adyacentes a la hormiga principal, en las adyacentes que ya haya descubierto y en las que estén descubiertas.

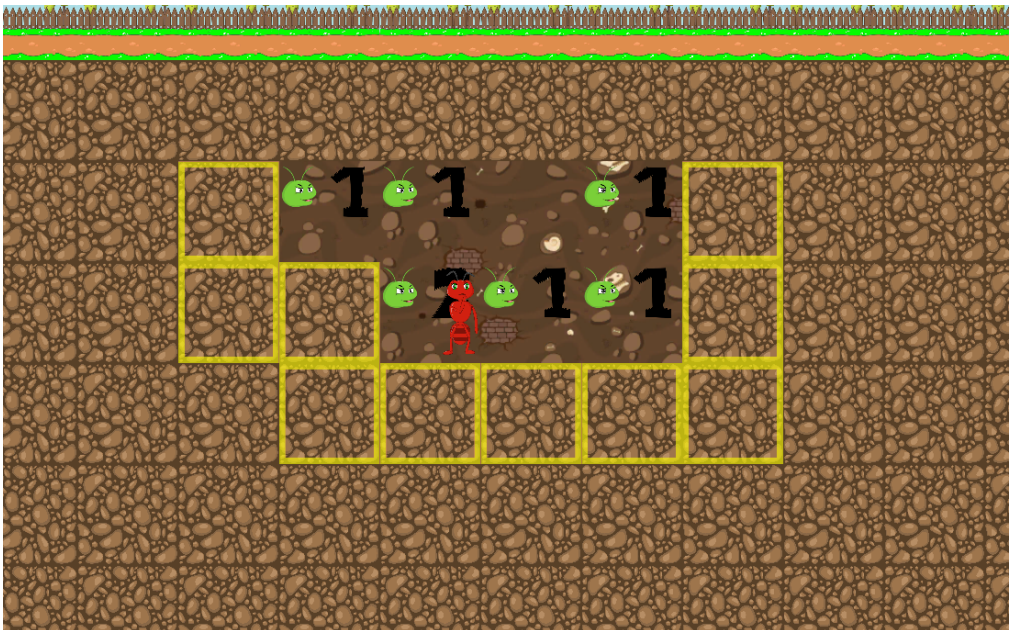


Ilustración 19 Celdas pulsables

Si la celda se puede pulsar, el juego te avisara marcándola con un rectángulo amarillo (ilustración 19). Aunque no este en amarillo, la hormiga si que podrá moverse por las celdas descubiertas.

Por otro lado, lo que nos podemos encontrar al destapar las celdas es:

- Celda vacía: Si así fuera descubriría las celdas adyacentes con el algoritmo Floodfill.
- Celda con enemigo: Podemos encontrar una cigarra, la lengua del oso hormiguero o un topo. Si así fuera marcaría el Game Over de la partida.
- Celda con una hormiga compañera: Añadiría puntos y si se consiguen todas, la victoria de la partida.
- Celda con aviso: Estas celdas pueden venir configuradas de distintas maneras. Lo que marca es que hay un enemigo cerca. Las configuraciones son las siguientes:
    - Aviso de cigarra: Te avisa de que hay una o varias cigarras cerca dependiendo del numero.
    - Aviso de topo: Te avisa que hay un topo cerca, en este caso solo te informa de la cercanía del topo no del numero de ellos.
    - Aviso de oso hormiguero: Te avisa de que la lengua del oso hormiguero esta próxima, al igual que el topo, te avisa de la cercanía no del numero de “trozos” de lengua.

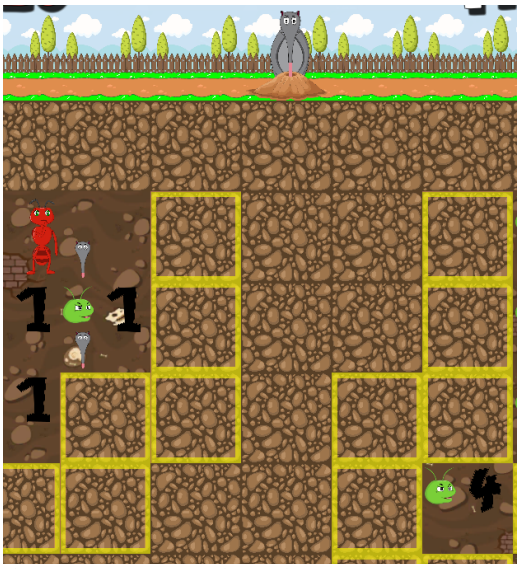


Ilustración 20 Marca de oso hormiguero cercano

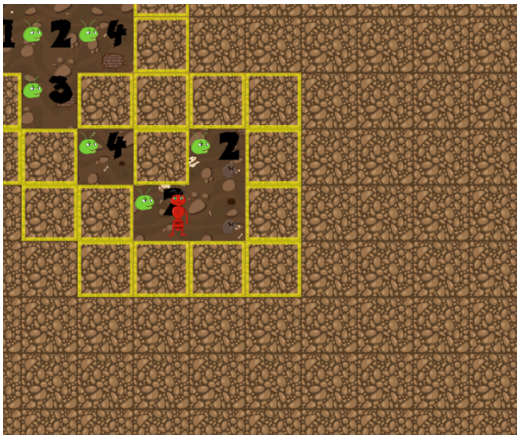


Ilustración 21 Marca de topo cercano

En último lugar, en cuanto al descubrimiento de celdas, hemos comentado anteriormente que si encontramos una celda vacía activamos el algoritmo floodfill. Este algoritmo creado por S. Fazzini, requiere tres parámetros: un nodo inicial, un color a sustituir y otro de relleno. El algoritmo rastrea todos los nodos que sean del color seleccionado, y a la vez contiguos entre sí y con el inicial, y los sustituye por el color seleccionado [1]. En nuestro caso, el nodo inicial es la celda pulsada, buscará celdas vacías y se parará al encontrar un aviso de enemigo, si no encuentra una celda de aviso continuará con las celdas vacías contiguas a esta. De esta manera, podremos ir destapando celdas con este algoritmo.

4.2 Puntuación

Como avanzamos anteriormente, una parte importante del juego es la puntuación, tenemos diversas formas de conseguir puntuación:

- Descubriendo celdas.
- Marcando enemigos, según el enemigo marcado la cantidad de puntos es distinta, los enemigos se marcan con señales (explicado en el apartado de objetos).
- Salvando hormigas, es la mayor cantidad de puntos que se puede obtener.

Los puntos se guardarán dependiendo del modo de juego seleccionado, por lo tanto, siempre que entremos al juego tendremos una puntuación máxima a batir dependiendo de modo de juego.

Otro punto importante en el videojuego es la victoria. Una vez finalizada la partida, tendremos una opción para poder continuar la partida y conseguir más puntos, es decir, que podremos continuar la partida incluso al conseguir todas las hormigas compañeras arriesgándonos a perderlo todo. Si quisiéramos finalizarla tendríamos un botón que nos permite finalizar la partida y entonces el juego guardará los puntos obtenidos.

4.3 Objetos

Para ayudarnos en la partida, el usuario dispone de una serie de objetos que podrá utilizar, sin embargo, estos tienen algunas limitaciones.

Los objetos principales del usuario son las señales (ilustración 15). Estos objetos nos sirven para marcar donde creemos que encontraremos un enemigo. Tenemos 3 señales, una por cada tipo de enemigo que encontramos en el juego y cada una de estas señales tiene 3 reintentos. Si acertamos en el lugar y tipo de enemigo no nos restará un reintento. Por otro lado, si falláramos nos restaría uno. En el caso de llegar a 0 no podríamos volver a usar este objeto, lo que dificultaría bastante la partida.

De la misma manera que las señales, las bombas (ilustración 14) nos pueden ayudar en nuestra partida, pero estas, una vez usada, restará uno siempre. Este objeto nos servirá para destapar 9 casillas de golpe en el sitio que queramos, no hace falta que la celda esté marcada como amarilla. Recordemos que siempre que destapemos una celda activaremos el algoritmo floodfill, por ello, destaparemos como mínimo 9 casillas, ya que podemos encontrar alguna vacía y activar una reacción en cadena. Por último, comentar que, si la bomba alcanza a algún enemigo, este quedará inhabilitado y el usuario no perderá la partida. Es el único momento donde los enemigos pueden ser visibles por el usuario.

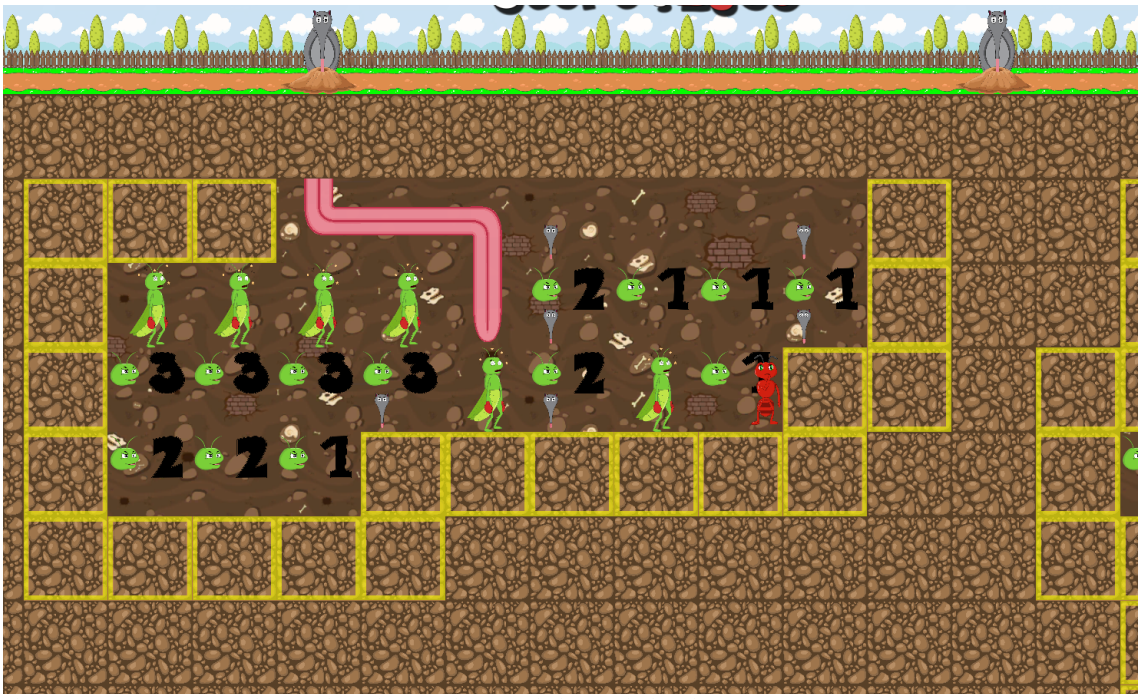


Ilustración 22 Enemigos derrotados y lengua despues de lanzar la bomba

4.4 Cámara

La cámara del juego es fija y siempre estará centrada en la hormiga principal. Excepto en algunos casos como límite de pantalla, donde dejará de estar centrada.

5. Diseño técnico

5.1 Entorno elegido

Antes de evaluar las diferentes opciones que tenemos, vamos a comentar las características principales del proyecto que me hicieron decantarme por un motor gráfico:

- Juego 2D de mecánicas muy sencillas
- Recursos gráficos muy simples
- No se exige una carga muy grande en pantalla
- Juego móvil, aunque también buscamos que sea fácil pasarlo a otros sistemas
- Tiempo limitado para realizar la tarea
- Fuerte carga lógica en la parte jugable del juego

Por lo anteriormente expuesto, buscamos un motor gráfico que nos ayude a ganar tiempo, con una curva de aprendizaje adecuada. Que admita la programación en diversos sistemas, entre ellos el móvil. Por otro lado, tiene que ser fácil tratar con recursos gráficos en 2D. Al ser la primera vez que programamos un videojuego el motor tiene que tener una comunidad activa

elevada con buenos tutoriales. Por último, y a nivel personal, tiene que ser un motor actual, ya que me gustaría aprender algo que fuera mas allá de la TFG.

Con todo ello, vamos a evaluar algunas de las opciones que he encontrado:

- **Unreal Engine:** Es un motor de videojuegos creado por Epic Games. El primer juego en usarlo fue el Unreal en 1998, en principio fue creado para shooters pero debido a su popularidad se ha expandido a otros géneros de juego. Ha pasado por diversas versiones la mas actual es la 5, pero aun no esta disponible. Por lo que sabemos, esta será compatible para la nueva generación de consolas. Es un motor muy potente, lo podemos usar libremente y esta muy bien optimizado.
  Pagina Web: https://www.unrealengine.com/

- **Game Maker:** Es una plataforma para desarrollar videojuegos creada por Mark Overmars en el lenguaje Delphi. Esta orientada para usuarios novatos con una interface muy sencilla e intuitiva. Esta especializado en desarrollos 2D como Hotline Miami, Undertale, etc. Tiene soporte en diversas plataformas e integración con Api de Steam, Amazon, etc.
  Pagina Web: https://www.yoyogames.com/gamemaker

- **Unity:** es un motor de videojuegos que nos permite renderizar gráficos 2D y 3D. Tiene un motor físico que nos permite simular las leyes físicas y una total libertad de parametrización de videojuegos gracias a su scripting en C#. El motor, al igual que el Unreal Engine, se puede usar libremente y tiene una comunidad muy activa.
  Pagina: https://unity.com/

Tenemos más opciones, pero estos fueron los motores gráficos que vi que encajarían mejor con el proyecto. Bajo mi punto de vista, Unity, aunque menos optimizado que Unreal Engine, es mucho mas accesible que este. Unity esta destinado a juegos con una carga gráfica menos exigente. En consecuencia, este motor sería mejor opción que Unreal Engine debido a las mecánicas y recursos simples de nuestro proyecto.

Además, Unity es un motor muy utilizado con una comunidad muy activa y sobretodo con proyección de futuro, en este punto está un escalón por encima que Game Maker. Por otro lado, descartamos Game Maker por que conozco menos Delphi que C# y la licencia gratuita de este es muy reducida.

Por todo ello**,** la elección para realizar el proyecto es Unity, la versión 2020.1.0f1.

5.2 Herramientas utilizadas

Dejando de lado los motores gráficos, pasamos a las herramientas complementarias para poder hacer posible el proyecto:

- Los scripts los realizaremos con el lenguaje de programación C#, para ello usaremos el **IDE Visual Studio**. Es un IDE para Windows y MAC compatible con diversos lenguajes de programación, por otro lado, es el editor de scripts predeterminado de Unity.
  Pagina: https://visualstudio.microsoft.com/es/

- La parte visual la realizaremos con **Inkscape**, es un editor de vectores gráficos gratuito. Al usar vectores gráficos no hace falta tener un gran conocimiento de dibujo para poder realizar trabajos vistosos. Por ello, este programa nos servirá para hacer tanto personajes como escenarios sencillos.
  Pagina: https://inkscape.org/es/

- Para el sonido usaré **Audacity**, programa gratuito que nos permitirá editar o grabar sonidos. Es un programa sencillo de usar que nos puede servir para este tipo de juegos.
  Pagina: https://www.audacityteam.org/

- Para llevar un control de versiones se ha utilizado el **Github,** gracias a este sistema se ha podido llevar un control a lo largo de la asignatura con subidas continuas desde el aplicativo **Github Desktop.**
  Pagina: https://desktop.github.com/

5.3 Scripts

En este capítulo desglosaremos los diferentes scripts que tiene el proyecto:

- **AnteaterManager:** Script encargado del funcionamiento del oso hormiguero, animaciones, sonidos, etc. Comentar también que este script tiene dentro un Array de TongueManager, ya que un oso hormiguero está compuesto de uno o varios "trozos" de lengua.
- **BombManager:** Script que maneja el sonido, la animación y el funcionamiento de la bomba.
- **CanvasManager:** Script que se encarga de controlar todos los Canvas del juego.
- **Cell:** Script de suma importancia ya que es con el que interactúa el jugador en la partida. Se encarga de controlar una celda. Este script controla todo lo que pasa en una celda, desde lo que tiene dentro mediante booleanos (enemigos, amigos, zonas vacías, etc.) hasta los métodos para descubrir que nos podemos encontrar al descubrirla.
- **CicadaManager:** Parte de código encargada de la cigarra. Su funcionamiento, animaciones, sonidos, etc.

- **FollowCamara:** Con este script podremos controlar la cámara, tiene asociado una serie de límites para no ir "mas allá" del propio nivel y, siempre que pueda, la cámara seguirá a la hormiga principal.
- **FriendManager:** Script encargado de las animaciones y sonidos de la hormiga a salvar.
- **GameManager:** El manager principal del juego, con este script realizamos las llamadas necesarias para crear un nivel y gestionar lo que pasa en la partida, desde sus modos de juegos hasta la validación del fin de partida o la victoria.
- **GenerateGame:** Script esencial del proyecto. Este script se encargará de generar el nivel, dependiendo del nivel de dificultad y una serie de porcentajes, creará toda la partida y enemigos. En capítulos posteriores explicaremos como realiza esta función.
- **LevelManager:** La función de este Script es la de gestionar el tablero entero, en este código nos encontraremos el algoritmo Floodfill, ya que es el único script que debe controlar todas las Cell de manera conjunta debido a que conoce el valor de cada una de ellas.
- **MenuManager:** Script que se encarga de controlar los menús del inicio del juego, engloba las opciones, elección de los modos de juegos, la llamada a la siguiente escena, etc.
- **MoleManager:** Script para controlar las animaciones, sonidos y funcionamiento del topo.
- **PlayerManager:** Para poder controlar a la hormiga principal tenemos este Script, donde encontraremos los métodos para sus animaciones y movimientos.
- **TongueManager:** Este Script hace referencia a cada uno de los "trozos" de lengua que pueden ocupar una celda, estos trozos de lengua forman parte de un oso hormiguero y en el código podemos encontrar el tipo de lengua que es (si es su parte final, recta o un borde) y la forma en la que se generan.

5.4 Imágenes, fondos y sprites

El proyecto se contiene diversas imágenes que tenemos en la carpeta "Sprites". En ella podemos encontrar los carteles y bombas:

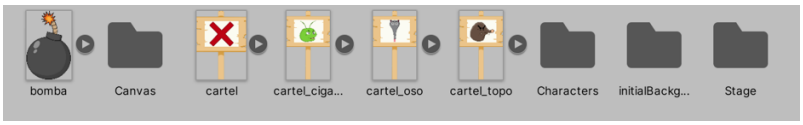


Ilustración 23 Carpeta Sprites

En la carpeta de "Stage" tenemos los elementos que componen un nivel con los números para indicar el numero de cigarras y el fondo de pantalla.

Ilustración 24 Carpeta stage

En "initialBackgroud" podemos encontrar la imagen principal del juego:

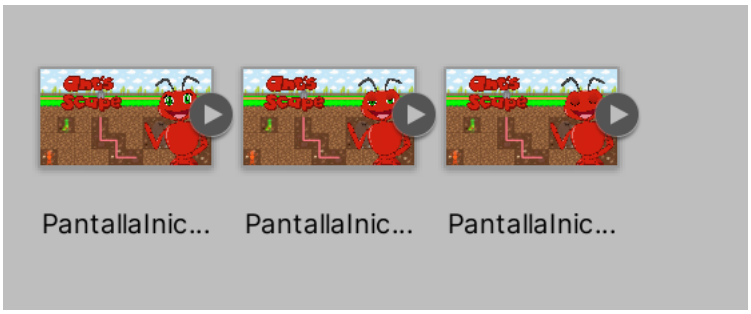


Ilustración 25 Carpeta initialBackgroud

En la carpeta “characters” tenemos una división para todos los elementos animados del juego:

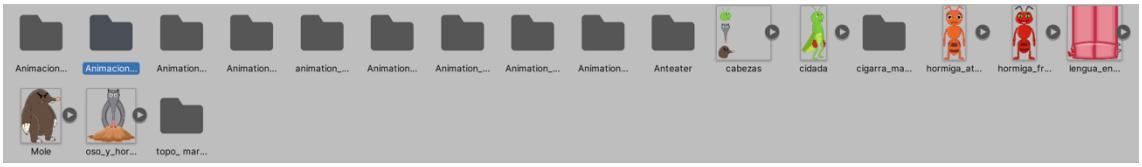


Ilustración 26 Carpeta Characters

Ilustración 27 Animación bomba

Ilustración 28 Animación oso hormiguero

Ilustración 29 Animación de muerte

Ilustración 30 Animación hormiga parada

Ilustración 31 Animación hormiga caminando

Ilustración 32 Animación cigarra ataque

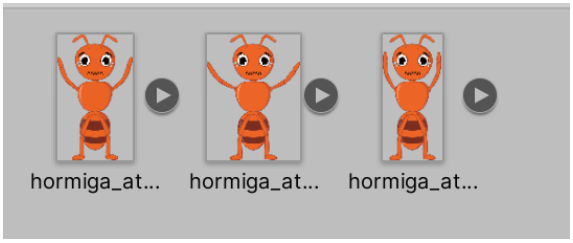


Ilustración 33 Animación hormiga miedo

Ilustración 34 Animación hormiga salvada

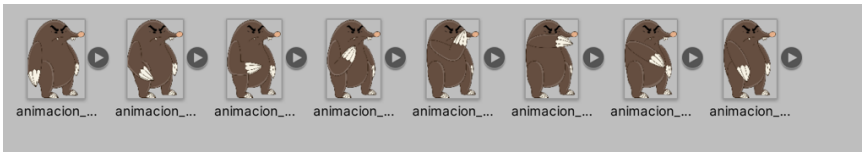


Ilustración 35 Animación topo ataque

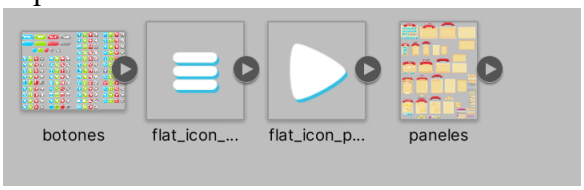
Ilustración 36 Cigarra derrotada

Ilustración 37 Topo derrotado

Dentro de la carpeta “characters”, también podemos observar la carpeta del “anteater” en la cual encontraremos la lengua del oso hormiguero:

Ilustración 38 Lengua

Por último, dentro de la carpeta “Sprites/Canvas” tenemos todas las imágenes de la parte UI.



Comentar que estas imágenes, junto con la música y sonidos, son los únicos assets que no están realizados por mi. Detallaremos de donde hemos obtenido los assets que no son míos, aunque son de libre uso:

- Brain Trust (música principal)→ Descargada de la biblioteca de Youtube:https://studio.youtube.com/channel/UCJbQV6DmWilae1NIh6C8dKQ/music
- Punch (sonido) → https://freesound.org/
- Bubble_button (sonido)→ https://freesound.org/
- Yujuu (sonido) → https://www.zapsplat.com/music/male-shout-woo-hoo/
- Game Over (sonido): https://freesound.org/
- Tadaa (sonido de victoria): https://freesound.org/
- Zarpazo (sonido): https://freesound.org/
- GUI (imágenes de la UI): https://opengameart.org/content/free-game-gui

5.5 Prefabs

Los prefabs son instancias de un objeto con unas propiedades definidas, de esta manera no tenemos que reprogramar propiedades ni duplicar objeto. Se han creado prefabs en la hormiga principal, oso hormiguero, bomba, celda, hormiga amiga, topo, cigarra y distintas configuraciones de la lengua del oso hormiguero.

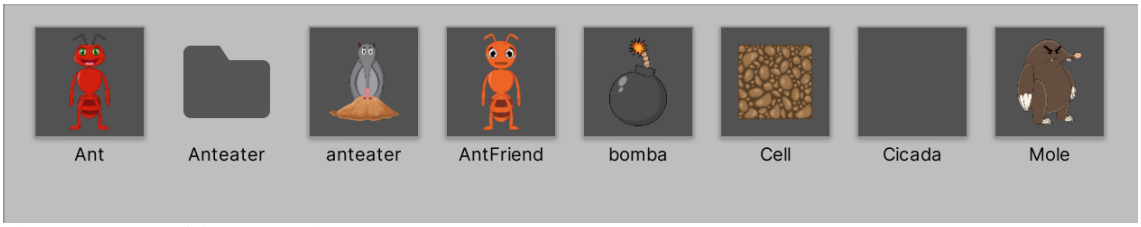


Ilustración 39 Prefabs principales

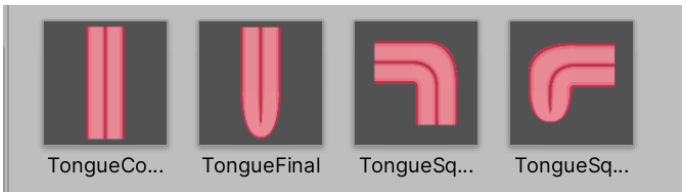


Ilustración 40 Prefabs de lengua

5.6 Música y sonido

Como hemos comentado anteriormente, la música no es propia, aunque todas pueden usarse libremente. Las encontraremos en la carpeta “sounds” dentro de “Assets”:

Ilustración 41 Sonidos del juego

6. Diseño de niveles y modos de juego

6.1 Generación de niveles

Los niveles son una parte esencial en el mundo de los videojuegos, en el presente proyecto los niveles se generan de manera aleatoria teniendo en cuenta la medida y la dificultad seleccionada por el usuario.

La generación de niveles se realiza una vez que el usuario selecciona el modo de juego y cambio de escena. Como avanzamos en capítulos anteriores, el encargado de generar el nivel es el script GenerateGame. El método Generate empezará creando un tablero de celdas de una longitud fija partiendo de la elección del usuario. Una vez creado el tablero, realiza las siguientes acciones:

- Generación de zona amiga donde no podrán aparecer enemigos. Por ejemplo, cuando generamos el prefab de la hormiga principal, también marcamos como zona amiga la casilla de su izquierda, derecha y abajo.
- Generación de las hormigas a salvar. Se crearán en casillas libres y, una vez instanciada la hormiga, sus casillas adyacentes serán marcadas como zona amiga para que no aparezcan enemigos en esas celdas.
- Se generan las celdas "no jugables" como la tierra que esta encima del tablero a modo decorativo.
- Se generan las cigarras en las celdas libres.
- El siguiente paso es la generación de un oso hormiguero, este tratamiento se compone de 2 bloques:
    - En primer lugar, el script lo que intenta es buscar una celda libre en la parte mas alta del tablero (la mas cerca de la tierra, ya que el oso hormiguero es visible) de manera aleatoria. Una vez encontrado un hueco libre, generará el enemigo en la parte de tierra del escenario y marcará la celda para saber que se va a generar un trozo de lengua.
    - El segundo paso, y una vez que tenemos generado el enemigo, será crear la lengua. La lengua se irá generando celda a celda con una probabilidad del 70%, si en algún intento de esa probabilidad sale falso será la parte de la punta de la lengua y daremos por finalizada la generación de la lengua de ese oso hormiguero. Cabe decir, que en la creación de la lengua puede tomar 3 direcciones, hacia la derecha, izquierda y abajo. Esta dirección la elige de manera aleatoria, teniendo en cuenta que la celda esta libre y teniendo en cuenta que si, de manera aleatoria, puede elegir entre derecha o izquierda y abajo, siempre ira hacia abajo.

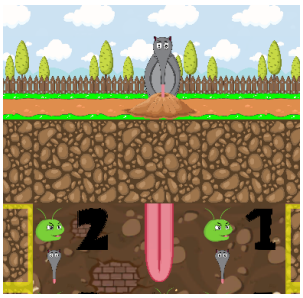


Ilustración 42 Topo y lengua

- El último paso de la generación del nivel es la creación de los topos. La instanciación de estos enemigos se realizará en la parte baja del tablero (de la mitad hacia abajo), el script buscará de manera aleatoria una celda, revisará que tenga otras 3 celdas adyacentes libres donde poder generar el topo y lo instanciará.

6.2 Dificultad y tamaño

El juego dispone de ciertas opciones para poder configurar el tamaño del tablero y la dificultad.

Ilustración 43 UI Elección dificultad y tamaño

Estas opciones afectan mucho al juego, evaluamos las posibles elecciones.

- Dificultad.
  - Fácil: Con esta opción el numero de enemigos se verá reducido:
    - Cigarra → 15% de salir en una celda.
    - Topos→No existen.
    - Osos hormigueros → No existen.
  - Normal: Opción por defecto del juego. El numero de enemigos es:
    - Cigarra→ 20% de posibilidades por celda.
    - Topos→ 2 topos como máximo.
    - Osos hormigueros→ 2 osos hormigueros como máximo.
  - Difícil: La opción mas complicada del juego, el número de enemigos es:
    - Cigarra → 28% por celda.
    - Topos→ 4 topos como máximo.
    - Osos hormigueros→4 osos hormigueros como máximo.
- Tamaño.
  - Pequeño: El mapa es reducido, ideal para una partida rápida, la configuración es la siguiente:
    - El tablero es 19x19.
    - Un máximo de 2 hormigas a salvar.
  - Normal: Opción en la cual el jugador necesitará un poco mas de tiempo para completarla, la configuración es la siguiente:
    - El tablero es 38x38.
    - Tenemos un máximo de 4 hormigas a salvar.
    - Multiplicamos por 2 el numero máximo de topos.
    - Multiplicamos por 2 el numero máximo de osos hormiguero.

- Grande: Opción para partidas largas y que requieren de bastante tiempo si se combina con el nivel normal o difícil, la configuración de este tamaño es la siguiente:
  - Un tablero de 76x76.
  - Tenemos un máximo de 16 hormigas a salvar.
  - Multiplicamos por 4 el numero máximo de topos.
  - Multiplicamos por 4 el numero máximo de osos hormiguero.

6.3 Modos de juego

El juego dispone de distintos modos de juegos a elección del jugador. A continuación, explicaremos en que consiste cada uno:

- **Personalizada**: Modo de juego de partida única donde el jugador escoge la dificultad y tamaño del tablero.
- **Infinito**: Modo de juego donde el jugador tendrá que enfrentarse a diversas pantallas hasta que pierda, la puntuación final será la suma que ha obtenido el jugador en todas las pantallas hasta ese punto. Las pantallas serán elegidas de manera aleatoria, sin embargo, las primeras serán pequeñas y de una dificultad reducida, poco a poco el jugador se encontrará con pantallas más grandes y se irá elevando la dificultad de estas. Otro punto a comentar es que entre pantalla y pantalla los objetos no se regeneran, por lo tanto, tendremos que administrar bien cómo y cuándo queremos usarlos.
- **Tiempo:** Este modo de juego tiene las mismas reglas que el modo infinito. Sin embargo, el final de la partida llegará al igual que los demás modos, al chocarnos con un enemigo, o al finalizar el tiempo 1200 segundos. Este modo esta realizado para alcanzar la máxima puntuación en un tiempo determinado.

7. Manual de usuario

7.1 Instalación

El proyecto, como ya hemos comentado anteriormente, esta desarrollado para móvil. Dentro del proyecto en el Github tenemos una carpeta llamada Ejecutables, dentro de ella podemos ejecutar el apk para instalarlo en un móvil Android o, como es mi caso, en un emulador de Android (BlueStacks):

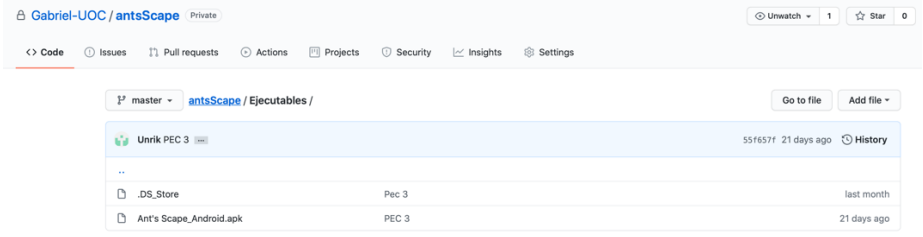


Ilustración 44 github ejecutable

7.2 Interfaces

El proyecto se compone de diversos canvas que componen toda la UI del juego. Cabe destacar que la composición de estos canvas, botones, imágenes, etc. no son propias, pero sí son recursos de libre utilización.

A continuación, vamos a explicar las diferentes interfaces que nos encontramos en el juego.

- Menú principal del juego: El menú se compone de un simple canvas con 4 botones, el botón de opción nos llevaría al menú donde configurar el sonido y el botón personalizada a otra pantalla donde configurar la partida.

Ilustración 45 UI Menu inicial

Ilustración 46 UI Opciones

Ilustración 47 UI Partida personalitzada

- UI dentro del juego: Para poder controlar ciertas opciones del juego tenemos la UI de la propia partida que nos permitirá pausar el juego, donde podremos salir o repetir la partida, y en la parte inferior podremos usar los objetos del juego, también visualizaremos la puntuación de la partida actual y el tiempo transcurrido, excepto en el modo de juego Tiempo que es una cuenta atrás.

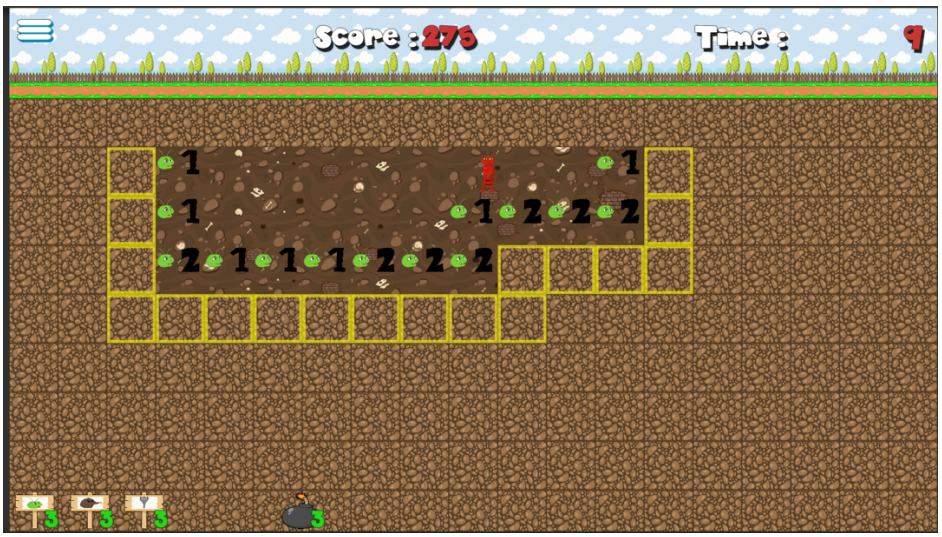


Ilustración 48 UI Partida del juego

Ilustración 49 UI Menu de pause

- UI fin de partida: En el juego también disponemos de diversas pantallas una vez finalizada la partida, como el Game Over o la pantalla de victoria, esta puede cambiar dependiendo si batimos el record o no.

Ilustración 50 UI Partida ganada

Ilustración 51 UI Game Over

7.3 Controles

Respecto a los controles del juego, estos son muy simples al ser un juego creado para móvil. El usuario solo podrá pulsar las casillas activas en la pantalla, realmente él no controla a la hormiga, esta únicamente "persigue" la acción.

Además, el usuario también podrá pulsar en los botones de la interface, incluidos los objetos que cambiaran de color dependiendo cual tengamos activo.

8. Mejoras del proyecto

El proyecto se inicio con las tareas comentadas en la planificación de tareas, sin embargo, en estos 3 meses de duración han surgido algunas ideas (como los modos de juego) que dio tiempo a desarrollar y otras que por no poner en peligro la finalización del proyecto no se realizaron.

Algunas ideas de mejoras también han surgido en la finalización del proyecto, en este punto se ve mucho mejor los puntos flojos del videojuego. Estos son algunos de los ejemplos para mejorar el producto:

- Dar la posibilidad de creación de niveles a medida en vez de hacerlos de manera aleatoria. De esta manera podemos tener algunos niveles realizados a mano, dando pie a la creación de una comunidad donde poder traspasarse los niveles generados.
- Con relación al punto anterior, cambiar la programación para que los niveles generados aleatoriamente sean mediante seeds, de esta manera, podremos guardar los niveles generados aleatoriamente o nos facilitara la creación de ellos. Gracias a ello, únicamente estaríamos guardando una

configuración de niveles para que sea mucho mas fácil crear la comunidad de niveles.

- Al agregar creaciones de niveles a medida, se puede crear un modo historia o mapa de niveles en los cuales el usuario tenga que ir superándolos uno a uno. Al igual que otros juegos, estos niveles pueden tener estrellas a batir según la puntuación obtenida, de esta manera aumentamos la rejugabilidad del videojuego.
- Se necesitaría hacer cambios en la puntuación, ya que también debería ir asociada al tiempo transcurrido en la partida y los objetos sobrantes, de esta manera el jugador tendrá mucho mas cuidado a la hora de utilizar objetos y, por otro lado, el tiempo tendrá un valor para el jugador.
- Se podría crear un ranking online y retos mensuales para crear competitividad entre jugadores.
- Este punto va destinado a un posible cambio en la jugabilidad, pero cambiaría mucho el juego. Por ello, lo dejaría para un modo de juego aparte. Este modo de juego el oso hormiguero pasaría a ser dinámico, en el presente proyecto la lengua del oso hormiguero se genera junto a la pantalla y su lengua no vuelve a cambiar durante la partida. Esto lo podríamos cambiar y hacer que la lengua creciera si el jugador descubre una casilla, de esta manera tendríamos un juego mucho más dinámico y obligaríamos al jugador a pensar más sus movimientos. En este punto, tendríamos que ajustar que la lengua creciera dependiendo de un porcentaje, 50% por ejemplo, por que si no fuera así podríamos desbalancear la dificultad.
- El último punto va destinado a nuevos objetos. Cómo, por ejemplo, un muñeco de hormiga, este objeto dejaría incapacitado el oso hormiguero. Este objeto viene muy ligado con el punto anterior, ya que al hacer dinámico el oso hormiguero aumentamos la dificultad del juego, por ello, debemos otorgarle algún arma al usuario para poder hacerle frente. Por ese motivo, el muñeco se lo podremos lanzar a la lengua del oso hormiguero y todos los trozos de lengua de ese enemigo desaparecerán y dejará incapacitado al oso hormiguero durante X turnos. Cuando este despierte, podrá sacar la lengua de nuevo, pero empezará desde el principio por lo tanto solo tendrá un trozo de lengua.

9. Viabilidad del producto

9.1 Sector destinado

Evaluando el mercado del videojuego podemos apreciar que mas de la mitad de los ingresos del mundo de los videojuegos provienen de los móviles [8]. Es una plataforma que ha tenido una explosión enorme a lo largo de los últimos años, como ya comentamos en capítulos anteriores tenemos como referencias Angry Birds o, más reciente, Candy Crush.

Por otro lado, también hemos comentado que el género de puzles, que es del que trata el proyecto, tiene un público muy amplio debido a que no requiere conocimientos previos del mundo de los videojuegos ni tiene mecánicas desconocidas para el usuario.

Por todo ello, creo que la fusión del género de lógica y el mercado móvil puede llegar a un público muy amplio

9.2 Monetización del producto

Cómo hemos comentado, el videojuego será para teléfonos móviles. En estos dispositivos tenemos diversas opciones de monetización. En nuestro caso, el juego será gratuito en su descarga, pero podemos introducirle publicidad cada 2 derrotas y micropagos en los objetos.

Cómo explicamos, el usuario tiene diversos objetos para poder ser usados en la partida. Por una parte, encontramos las bombas que únicamente tiene 3 y no se recargan en la propia partida. También dispone de las señales, las cuales si falla al colocarlas perderá una. Podríamos poner compras en la misma partida para que si nos quedamos sin objetos el usuario pueda comprarlos si así lo desea.

Como último punto, podríamos otorgarle una vida al jugador, es decir, si el jugador llega al Game Over, gastar una vida previamente comprada o incluso tener, como tienes otros muchos juegos, 5 vidas como máximo y al gastarlas esperar un tiempo, comprarlas o ver algunos anuncios. Como podemos comprobar tenemos muchas formas para poder monetizar el proyecto.

10. Conclusiones

Llegando al punto final de la TFG no me arrepiento de haber elegido el camino de los videojuegos. Ha sido una gran experiencia y un desafío introducirme en un mundo que cada día evoluciona, llegando incluso hoy en día a superar los ingresos del cine [9].

Pero esta evolución no ha sido casualidad, ha sido fruto de grandes logros. Cómo, por ejemplo, el cambio de mentalidad que asociaba videojuegos a niños, el auge del juego móvil o el gran avance tecnológico, que hace que parezca que estemos ante una película de animación en vez de un videojuego. A todo ello, le tendríamos que sumar el renacimiento de los juegos de PC y los desarrollos de los juegos indies, que han dado pie a realizar juegos con un prepuesto muy bajo y realizados por un grupo de personas reducido, incluso pudiendo ser una sola persona, Stardew Valley por poner un ejemplo.

Con el paso del tiempo, más gente se siente atraída por esta industria en plena evolución, no solo por ocio si no también por oportunidades laborales. Algo que he aprendido durante el proyecto es que al realizar un videojuego es un producto que requiere el conocimiento de muchos profesionales de distintas áreas: dibujantes, modeladores, músicos, programadores, guionistas, etc.

Otra de las lecciones aprendidas es el tiempo que requiere realizar un videojuego. Esta es una tarea muy compleja, por muy simple que sea la idea es algo que requerirá mucho tiempo a alguien con poca experiencia, como es mi caso. El tiempo consumido en los assets, como los sprites y las animaciones, ha sido mucho mas de lo que imaginé.

En referencia al punto anterior, también he aprendido a valorar el tiempo de otra manera. Teníamos muy poco tiempo para realizar el producto, pero creo que el proyecto necesita un tiempo de “maduración”. Aunque la finalidad del videojuego era clara, una vez implementadas ciertas funciones, observaba algunos problemas en el plano jugable, debido tanto a mi poca experiencia como a la falta de tiempo. Por ello, creo que estos proyectos necesitan un proceso de análisis y planificación de tareas con antelación, además tienes que estar preparado para imprevistos como una funcionalidad que una vez implementada no funcione de la manera que esperabas. Por esta razón, se deberían implementar funcionalidades una a una e intentar tener un feedback sobre ellas para poder ir mejorando poco a poco el juego.

El proyecto se ha podido finalizar y ahora tenemos el producto acabado, alcanzando y logrando los objetivos planeados inicialmente. Aunque durante el desarrollo surgieron nuevas ideas que fuimos implementando, como los modos de juego o el uso de objetos, me hubiera gustado implementar el ranking online. Desgraciadamente las tareas de la memoria, documentación final y cambios para optimizar el código me han llevado mas tiempo del esperado.

En lo que respecta a la planificación se ha cumplido correctamente, sin embargo, algunas de las tareas las subestimé. Crear a los personajes y sus animaciones es

una tarea muy compleja para alguien con poca experiencia, el dibujo requiere tiempo y crear una animación para que se vea fluida y agradable en el juego fue algo que no tenia previsto, teniendo en cuenta que algunas animaciones las tuve que desechar por que no parecían naturales. Por otro lado, aunque en menor medida, valoré en muy poco tiempo la interface del videojuego. Esta tuvo 2 versiones, una mas simple realizada por mi y otra donde utilizamos assets gratuitos. Sin contar el funcionamiento de la interface que también me llevó mas tiempo del previsto.

Por último, comentar que las mejoras ya las hemos comentado en capítulos anteriores y, como dije, me parece que el producto tiene mucho margen de mejora sobre todo en ciertos aspectos de la parte jugable y en la creación de una comunidad online.

11. Glosario

Definición de los términos y acrónimos más relevantes utilizados dentro de la Memoria.

Script: Es un archivo que contiene una serie de instrucciones escritas en un código de programación que ejecuta diversas funciones en el interior de un computador.

C#: Lenguaje de programación desarrollado por la empresa Microsoft, su sintaxis básica deriva de C/C++ y el modelo de objetos de la plataforma .NET, similar a Java.

Algoritmo: Conjunto ordenado de operaciones sistemáticas que permite hacer un cálculo y hallar la solución de un tipo de problemas.

Interface: Mecanismo o herramienta que posibilita una comunicación mediante un conjunto de objetos, iconos y elementos gráficos.

UI: Estas siglas hacen referencia a la interfaz del usuario, que es la vista que permite a un usuario interactuar con el sistema.

Floodfill: Es un algoritmo de relleno en programas de dibujo, en castellano se llama algoritmo de relleno por difusión. Es utilizado en el buscaminas, Puyo Puyo, Lumines y Magical Drop para determinar qué piezas pueden retirarse o seleccionarse.

2D: Termino utilizado en los videojuegos para referirse a las dimensiones en las cuales se juega. En este caso 2 dimensiones.

3D: Termino utilizado en los videojuegos para referirse a las dimensiones en las cuales se juega. En este caso 3 dimensiones.

Assets: Es una representación de cualquier ítem que puede ser utilizado en un juego o proyecto.

Sonidos fx: Son los sonidos de efectos, que no forman parte de la banda sonora si no que representan el sonido natural de los elementos.

Fase beta: Es una fase del proyecto que ya ha alcanzado cierta madurez y se tiene la primera versión completa conforme a los requisitos iniciales.

Prefabs: Los prefabs son objetos reutilizables, creados con una serie de características dentro de la vista proyecto, que serán instanciados en el videojuego cada vez que se estime oportuno y tantas veces como sea necesario.

Motor grafico: Herramienta para crear y desarrollar un videojuego.

Indie: Un termino que significa independiente o independencia, en referencia a los videojuegos es la creación de un juego sin el apoyo financiero de una distribuidora de videojuegos.

Sprite: Son un tipo de mapa de bits dibujados en ordenador para ser usados en un videojuego y poder crear los gráficos.

API: Interfaz de programación de aplicación, es una biblioteca de funciones y procedimientos para ser utilizada por otro software.

Shooters: Es el género de videojuegos que pertenecen al genero de acción, en el cual el personaje del juego dispone de un arma que puede ser disparada

IDE: Las siglas significan entorno de desarrollo integrado que es un sistema de software para el diseño de aplicación que combina diferentes herramientas de desarrollo en una sola interfaz gráfica.

Emulador: Es un software que imita al hardware o sistema operativo que el objetivo final de ejecutar un programa, aplicación, software, etc.

BlueStacks: Emulador de Android.

Android: Sistema operativo móvil basado en núcleo Linux y otros softwares de código abierto.

Canvas: En el unity, canvas es el área donde todos los elementos UI deben estar.

Seeds: Son clases, de programación orientada a objetos, en las que tendrás el código de los datos o configuración para definir una tabla, pantalla, etc.

12. Bibliografía

[1] Algoritmo de relleno por difusión. Wikipedia [en línea] <https://es.wikipedia.org/wiki/Algoritmo_de_relleno_por_difusi%C3%B3n>

[2] Videojuego de lógica. Wikipedia [en línea] <https://es.wikipedia.org/wiki/Videojuego_de_l%C3%B3gica>

[3] Lo complejo y divertido: sobre los puzzles [en línea] <https://atomix.vg/lo-complejo-y-divertido-sobre-los-puzzles/>

[4] Buscaminas. Wikipedia [en línea] <https://es.wikipedia.org/wiki/Buscaminas>

[5] Lemmings. Wikipedia [en línea] https://es.wikipedia.org/wiki/Lemmings

[6] Angry Birds, el secreto detrás del éxito y qué aprender de sus creadores [en línea] <https://www.vidaextra.com/eventos/angry-birds-el-secreto-detras-del-exito-y-que-aprender-de-sus-creadores-gdc-2011>

[7] Candy Crush Saga. Wikipedia [en línea] <https://es.wikipedia.org/wiki/Candy_Crush_Saga>

[8] Más de la mitad de los ingresos del sector de los videojuegos ya proceden del móvil [en línea] <https://www.xatakamovil.com/mercado/mitad-ingresos-sector-videojuegos-proceden-movil-superdata>

[9] Los videojuegos ya generan más dinero que el cine [en línea] <https://www.qore.com/noticias/68292/Reporte-Los-videojuegos-ya-generan-mas-dinero-que-el-cine>