

**CREACIÓN DE VIDEOJUEGOS 2D CON UNITY**

Scarf: Rogue-like, generación procedural y redes peer-to-peer.

*Ciclo Formativo de Grado Superior de Desarrollo de Aplicaciones Multiplataforma*

**PROYECTO FIN DE GRADO**

*Autor: Mario Heras Peña*

*Tutor: Rubén Escobedo Gutiérrez*

*Agradecimientos*

Índice

1. ***INTRODUCCIÓN***

Con este proyecto se pretende profundizar un poco en el proceso de creación de un videojuego con el motor Unity.

El proceso de creación de un videojuego es un proceso costoso, exhaustivo y que puede ser expandido infinitamente si queremos. En este mundo conviven dos visiones, el videojuego como negocio y el videojuego como arte.

La industria del videojuego es una de las que más capital mueve a día de hoy y hace tiempo que ha superado en beneficios a la del cine por muchísima diferencia.

Dicho esto, ¿es posible ganarse la vida haciendo videojuegos por cuenta propia?

Probablemente no, pero eso es un tema aparte.

Profundizando en Unity, es un motor de videojuegos muy versátil y es uno de los más usado junto a Unreal Engine, pero, sobre todo, el más usado para los videojuegos *Indie*.

Unity permite la creación de videojuegos tanto 3D como 2D (este último siendo el elegido para este proyecto), utilizando diversas herramientas que nos ofrece este motor como: el editor de escena, manejo de cámara, editor de sprites, animaciones, audio y scripting en C#, entre otros.

Para terminar, este proyecto busca ir un poco más allá que simplemente la creación de un videojuego. Las ideas más importantes y destacadas de este proyecto son dos.

La generación procedural del escenario y un netcode peer-to-peer para jugar multijugador.

Antes de hablar en profundidad de estos temas, comentar que este proyecto personalmente tiene como principales motivaciones el poner en práctica conocimientos e ideas, ver cómo difiere crear un videojuego a crear otro tipo de aplicación, disfrutar con amigos del juego y sobre todo aprender.

***1.1 GENERACIÓN PROCEDURAL***

La generación procedural fácilmente explicada: en vez de crear manualmente el escenario, un algoritmo lo crea por nosotros, haciendo que cada escenario sea aleatorio, y, por consecuencia, ninguna partida será igual que la anterior.

Hay muchos juegos, sobre todo del género *rogue-like* que aplican esta técnica para lograr la rejugabilidad y aumentar la complejidad.

También hay muchas formas de aplicar este concepto, como por ejemplo el caso de *Minecraft*, el cual genera una *seed* o semilla aleatoriamente y eso dictaminará el terreno del mundo, elevaciones, montañas, ríos, mares...

-- TODO Imagen minecraft

En nuestro caso haremos algo más en pequeña escala, es decir, generaremos habitaciones o salas interconectadas y que todas estas juntas formen un pequeño nivel. Un gran ejemplo e inspiración es *The Binding of Isaac*.

-- TODO Imagen Mapa Isaac

-- TODO Completar esta sección

***1.2 PEER TO PEER***

Usualmente abreviado P2P, Peer to Peer es una arquitectura de red en la que cada usuario actúa tanto de cliente como de servidor.

-- TODO Imagen P2P

Dicho esto parece evidente el por qué escoger esta arquitectura para un proyecto pequeño. Costear un servidor que aloje partidas pequeñas de entre 2 a 4 jugadores no es rentable.

Por suerte Unity tiene un netcode muy bueno y reciente, el *Netcode for GameObjects*, que por defecto utiliza la arquitectura P2P.

-- TODO Completar esta sección

1. ***OBJETIVOS***

Hay varios objetivos con este proyecto.

El mundo de la creación de videojuegos es un terreno fascinante y en constante evolución, el cual requiere plantearse retos complejos y superarlos constantemente.

Considero que aprender a trabajar con Unity es una de las formas más lógicas de entrar a este mundillo o por lo menos tener unas pequeñas primeras impresiones de cómo funciona todo, además, Unity trabaja con C#, un lenguaje con mucho potencial y con el que me siento muy cómodo.

El objetivo más importante es aprender a utilizar Unity de forma eficiente y autosuficiente (menos tutoriales de YouTube y más tiempo en el programa), además de aprender las capacidades y posibilidades (adelanto que son prácticamente infinitas) de este motor.

El otro objetivo es sinceramente pasármelo bien, si te apasionan los videojuegos, crear uno es prácticamente igual de divertido, solo que muchísimo más costoso y requiere muchos conocimientos y tiempo dedicado. Además, me gustaría seguir ampliando este proyecto y jugar con mis amigos, eso es algo importante, ¿para qué vas a hacer algo y luego no usarlo?

Por último, pero no menos importante, crear un proyecto **de verdad** desde 0.

Investigación, planificación, análisis, diseño, desarrollo, implementación, pruebas.

***2.1 PLANIFICACIÓN***

En general todo este documento habla sobre la planificación de este, pero quería hacer especial mención a los requisitos obtenidos, a la planificación del trabajo, con diferentes esquemas y tableros.

Empezaremos con Trello, una herramienta de gestión de proyectos y tareas muy cómoda.

-- TODO Completar sección (Trello)

-- TODO Imagen Trello

Además de esto, se ha hecho una planificación en horas para el proyecto.

-- TODO Diagrama Gantt

1. ***FASES DEL PROYECTO***

-- TODO Completar esta sección

***3.1 DISEÑO***

-- TODO Completar esta sección

***3.2 IMPLEMENTACIÓN Y CONFIGURACIÓN***

-- TODO Completar esta sección

***3.3 PRUEBAS***

-- TODO Completar esta sección

***4. AMPLIACIÓN Y POSIBLES MEJORAS***

Ya he destacado anteriormente que un juego tiene una posibilidad de mejora y ampliación infinita.

Para empezar, hay muchísimos apartados los cuales pulir, la estética, el sonido, el diseño de enemigos, las posibilidades de combate...

Ya que este proyecto no está cerrado, en un futuro se prevé:

-- TODO Completar esta sección

***5. CONCLUSIÓN***

-- TODO Completar esta sección

***6. BIBLIOGRAFÍA***

-- TODO Completar

Juegos: The Binding Of Isaac, Enter the Gungeon, Nuclear Throne