Centro de Estudios de Postgrado

Máster en Profesorado de Enseñanza Secundaria Obligatoria, Bachillerato, Formación Profesional y Enseñanza de Idiomas

Universidad de Jaén

Centro de Estudios de Postgrado

Trabajo Fin de Máster

MOTORES GRÁFICOS

Alumno/a: **Morales García, Álvaro**

Tutor/a: Prof. D. Pedro González García
Dpto: Departamento de Informática

Índice

Índice de Figuras

Índice de Tablas

Resumen

Este Trabajo de Fin del Master tiene como objetivo el estudio de los motores gráficos y el desarrollo de una Unidad Didáctica dentro de este ámbito.

Hasta hace poco, el estudio y desarrollo de motores gráficos se centraba exclusivamente en la creación de ocio desde la industria de los videojuegos, Aunque los motores gráficos siguen teniendo esa función, hoy en día el uso de estos se ha diversificado llegando a otros campos dado el potencial de estos.

La Unidad Didáctica “Motores gráficos” se encuadra dentro del módulo de “Programación multimedia y dispositivos móviles (0489)”, el cual se enmarca dentro del Ciclo Formativo de Grado Superior de Desarrollo de Aplicaciones Multiplataforma.

Palabras clave: Motores gráficos, Videojuegos, Dispositivos móviles, Unity, Unidad didáctica.

Abstract

This Master's Final Project aims to study graphic engines and develop a Didactic Unit within this field.

Until recently, the study and development of graphic engines focused exclusively on the creation of entertainment from the videogame industry. Although graphic engines continue to have that function, today their use has diversified, reaching other fields due to their potential.

The Didactic Unit “Graphic Engines” is part of the “Multimedia Programming and Mobile Devices (0489)” module, which is part of the Advanced Vocational Training on Multi-platform Application Development.

Keywords: Graphics engines, Videogames, Mobile devices, Unity, Didactic unit.

Estudio Epistemológico

En este apartado se desarrollará el estudio epistemológico sobre el tema elegido, “Los motores gráficos”, que tendría relación con el tema “74. Sistemas multimedia” del temario de oposiciones de Informática del cuerpo de Profesor de Enseñanza Secundaria según la Orden de 1 de febrero de 1996 por la que se aprueban los temarios que han de regir en los procedimientos de ingreso, adquisición de nuevas especialidades y movilidad para determinadas especialidades de los Cuerpos de Profesores de Enseñanza Secundaria y Profesores Técnicos de Formación Profesional.

Este estudio epistemológico se estructura de la siguiente forma: Comienza con un apartado de introducción al concepto de motor gráfico y su relación con el sistema gráfico. Seguidamente, se verá un pequeño repaso a la historia de los sistemas gráficos y el nacimiento de los motores gráficos. Después de esto, se explicará un breve ejemplo que expondrá de forma resumida cómo usar un motor gráfico. En el siguiente apartado, se desarrollará el estado actual de la temática elegida, comentando qué funciones tiene actualmente un motor gráfico, sus aplicaciones y uso, así como qué motores gráficos tenemos disponibles hoy en día. A continuación, en otro apartado se comentarán las tendencias futuras de esta temática. Finalmente, se desarrollarán unas conclusiones obtenidas como resultado del estudio del tema elegido para este trabajo.

1. Introducción

Un motor gráfico o motor de videojuegos es un software diseñado para crear y desarrollar videojuegos. Estos cuentan con diversas herramientas que permiten al desarrollador sacar todo el partido de ese motor, así como dar rienda suelta a la creatividad de este.

Un motor gráfico suele contar con un motor de renderizado, un motor de físicas, para colisiones, así como otras herramientas que permiten incluir sonidos, música, animaciones, ejecución de varios hilos, gestión de memoria, etc. Estos elementos lo veremos en profundidad más adelante en este estudio (3.1.Arquitectura del motor gráfico). Además de esto, los motores gráficos ya cuentan con un sistema de comunicación en red que puede ser necesario para juegos multijugador.

Hoy en día hay gran cantidad de motores gráficos disponibles con múltiples funciones y utilidades. En este estudio veremos algunos de los que más se usan actualmente (3.2.Motores gráficos más utilizados), así como el uso de alguno de ellos para la creación de un videojuego (3.3.Uso del motor gráfico).

Si bien es cierto que un motor gráfico o motor de videojuegos se centra precisamente en lo que su nombre indica, la creación de videojuegos, el desarrollo y estudio de los motores gráficos ha hecho que su potencial sea tenido en cuenta más allá de la industria del videojuego. Siendo así, veremos cómo los motores gráficos han diversificado su uso y la importancia de esto (4.Tendencias futuras).

Dicho esto, esta parte de este Trabajo de Fin de Máster se centrará en el estudio de los motores gráficos como uno de los grandes pilares centrales dentro de lo que sería el campo de la computación gráfica.

2. Antecedentes

Es necesario conocer algo de la historia de los videojuegos y los sistemas gráficos para saber en qué punto nos encontramos actualmente en el ámbito de los motores gráficos.

A continuación, se explicarán algunos eventos que marcaron el desarrollo de los sistemas y motores gráficos a lo largo de la historia (Varonas et al., 2012; Villagrán, 2016) y se mostrará una línea temporal resumiéndolos en la Figura 1.

Alexander Shafto Douglas crea el primer videojuego, un "tres en raya" para el gigante computador EDSAC, en el año 1952. Nos encontramos entonces con lo que sería el comienzo del ordenador al servicio del ocio humano.

Poco después, a finales de los años 50, se crea por primera vez una imagen figurativa digital siendo esta una chica Pin-Up representada en el sistema gráfico SAGE (Semi Automatic Ground Environtment). SAGE sería uno de los primeros sistemas gráficos y se usó para procesar datos recogidos por el radar y localizaciones con ayuda del equipo TX-0.

William Higginbotham hace historia en 1958 con la creación de su videojuego Tennis for two. Esta máquina emulaba lo que sería un partido de tenis con la ayuda de un osciloscopio, que serviría de pantalla y dos mandos, cada uno de ellos con un botón para golpear la bola y una rueda para controlar su dirección.

En el año 1963, Ivan Sutherland crea una interfaz gráfica para diseñar objetos en un ordenador siendo pionero en la representación de gráficos digitales con uno de los primeros programas de diseño asistido para un equipo informático, Sketchpad. Sería el origen de las pantallas táctiles actuales ya que utilizaba un lápiz para dibujar en la pantalla usando también algunos interruptores para insertar ciertas formas.

Se debe hacer una mención especial a Douglas Engelbart, ingeniero e inventor, que, junto a su amigo Bill English, crearía el primer prototipo de un dispositivo cuya función sería "apuntar" a los elementos dibujados en una pantalla en 1964. Nace así el mouse o ratón.

En 1974, James F. Blinn comienza a estudiar y profundizar en la representación gráfica realista, consiguiendo resultados que se convertirían en técnicas que se utilizan hasta el día de hoy como los modelos de iluminación especular, efectos ambientales, reflexión de objetos y el "bump mapping".

Unos años más tarde, en 1979 se creaba el primer vídeo con efectos 3D de la mano del artista visual Ed Emshwhiller. Este vídeo duraba unos 3 minutos y fue creado con Paint.

Entre 1983 y 1988 se crearon lo que serían algunos “kits de creación de juegos”, como Adventure Construction Kit (1984) y Garry Kitchen’s GameMaker (1985) entre otros. No llegaban a ser motores gráficos, pero se acercaban a la idea que tenemos ahora de un motor gráfico

Aparece Ultima Underworld en 1990, por parte de la compañía Origin Systems, un juego que contaba con un motor gráfico propio, siendo el primero que permitía el mapeado de texturas.

Voxel Space se lanzaría en 1992 mezclando en su desarrollo dos conceptos claves en el mundo de los motores gráficos: volumen y pixel. Se introdujo entonces el concepto de “Voxel”, lo que hizo que los gráficos quedasen mejor detallados y con una respuesta más rápida.

Poco después, comienzan a expandirse los first-person shooter o FPS al empezar a hacerse popular los juegos con gráficos “tridimensionales”. Con esto llega el juego DOOM (1993), que también traía su propio motor gráfico, el id Tech 1. Además, ese mismo año Build aparece en escena siendo el primer motor en soportar paredes en ángulo.

Nace Quake en 1996, motor gráfico con el que llega el 3D real. Este sería el motor que lo cambiaría todo, pues se introducía ya el Z-buffering, mejora de luces, prerrenderizado y reducción de polígonos por escena. Además, se introdujo la tecnología TCP/IP, por lo que empezarían a surgir los servidores dedicados y, con ello, nacería el gaming profesional.

Quake Engine 2 o id Tech 2 nace en 1998 siendo el primer motor que soportaría OpenGL de forma nativa. Ese mismo año también se crea Unreal Engine, la primera versión del tan afamado motor gráfico de la empresa Epic. Contaba con su propio lenguaje de programación, UnrealScript, y también un editor de código, un programa de modding para modificar los aspectos de los juegos y un creador de mapas (UnrealEd).

Tres años después, en 2001, Llega GeoMod un motor capaz de modificar la geometría de la escena según lo que ocurría en ella. Por ejemplo, podría modificar la geometría de la escena a causa de una explosión, derrumbando edificios y dejando las grietas en ellos.

Llegamos a 2005, año en el que nace Unity, el motor actualmente preferido por la mayoría de desarrolladores de videojuegos independientes, siendo este un motor muy barato y fácil de usar.

A partir de entonces surgen otros motores y nuevas versiones de grandes motores mencionados como Unity y Unreal Engine, los cuales proponen grandes mejoras en gráficos y acercándose mucho más a la realidad.

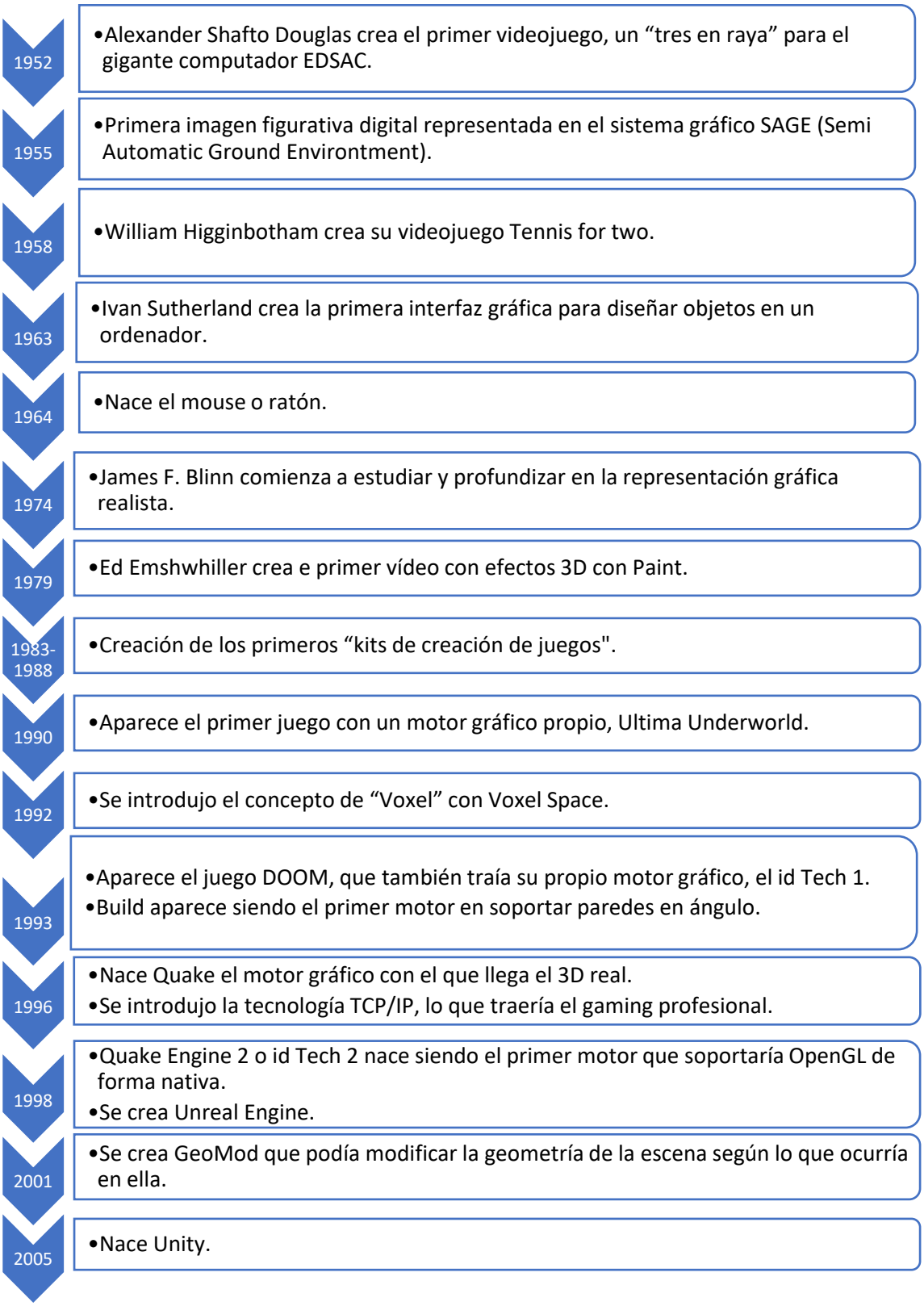

Figura 1: Historia de los motores gráficos. Fuente: Elaboración propia

3. Estado actual

El motor gráfico o motor de videojuegos es un conjunto de herramientas que facilitan la tarea de desarrollar videojuegos (Figura 2). Como comentábamos anteriormente, un motor gráfico se crea con una arquitectura que permite la reutilización de código haciendo una separación entre los diferentes componentes y herramientas que el motor gráfico nos ofrece, como el sistema de audio, el sistema de renderizado o el sistema de físicas (González et al., 2015).

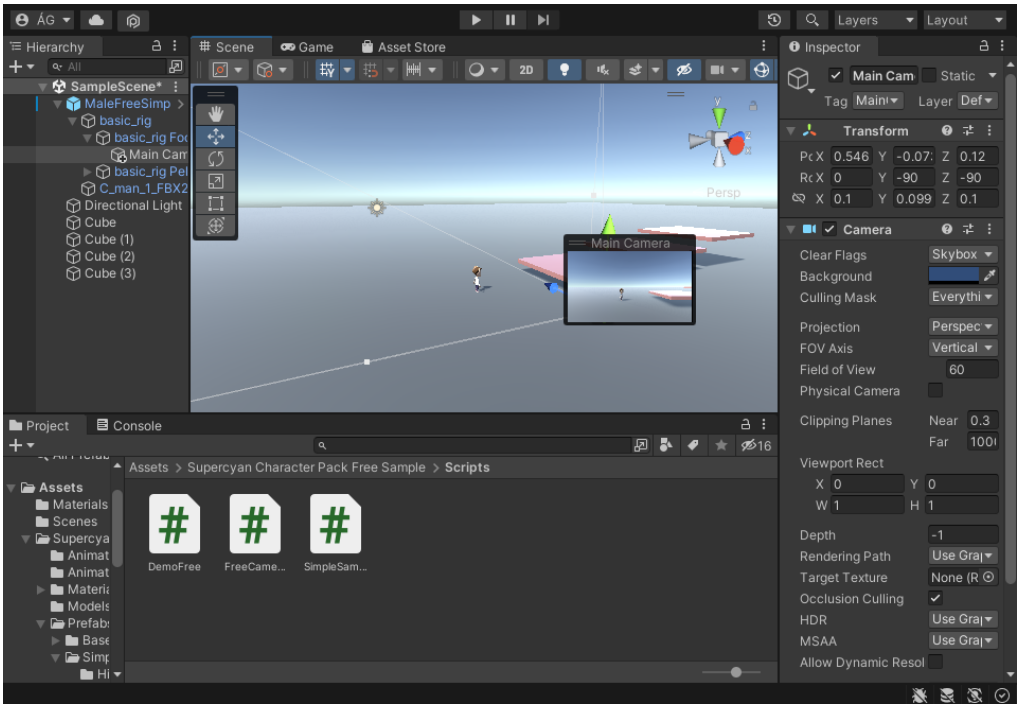

Figura 2: Ejemplo de motor gráfico. Fuente: Elaboración propia

3.1 Arquitectura del motor gráfico

Un motor gráfico o de videojuegos se organiza en bloques que se pueden utilizar según convenga en cada momento. En la Figura 3 se pueden apreciar estos bloques dentro de la arquitectura de un motor gráfico.

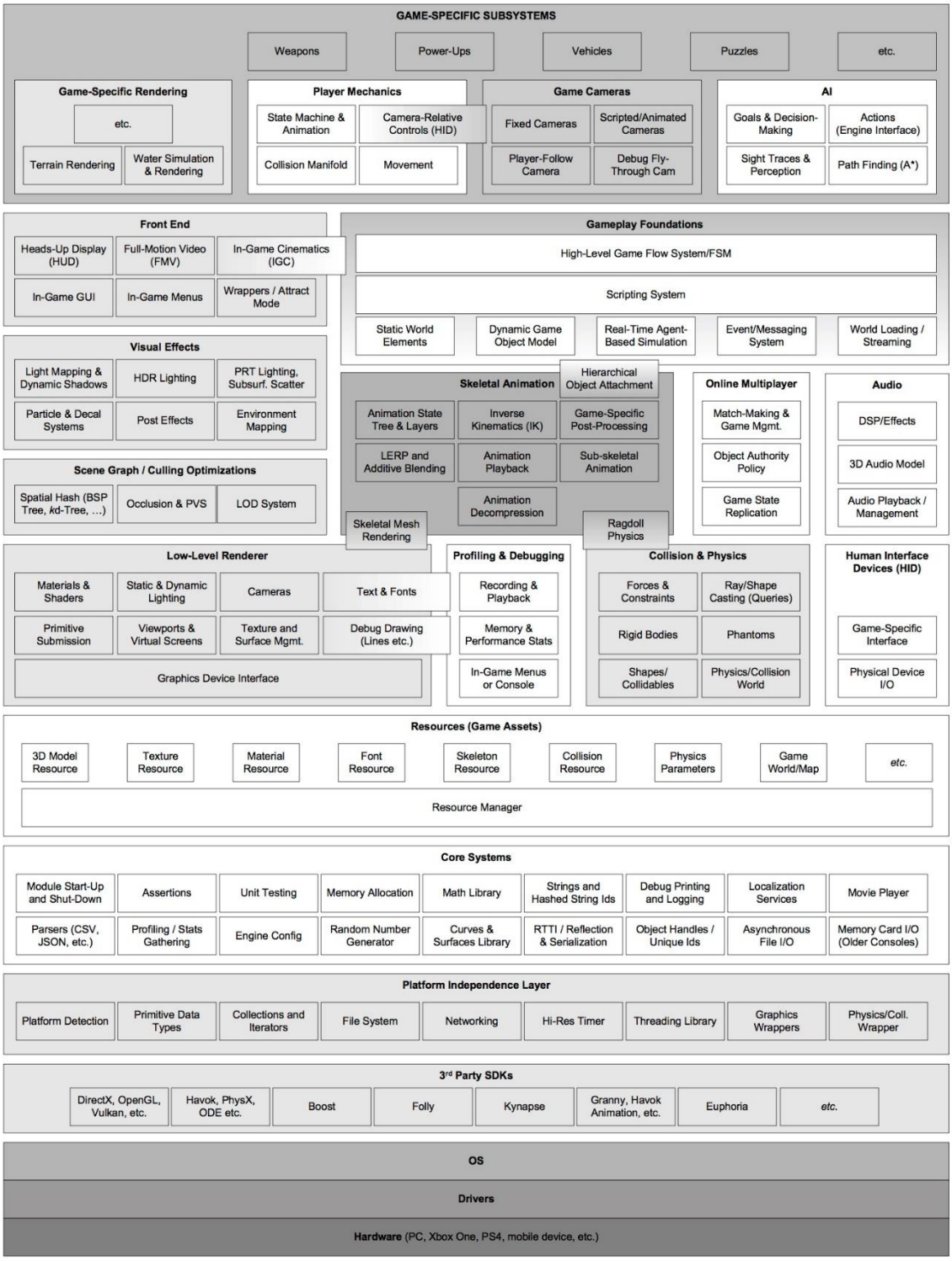

Figura 3: Arquitectura del motor gráfico. Fuente:(Gregory, 2009)

Como se puede ver, este gran sistema se compone también de capas que pueden ser interdependientes.

A continuación, veremos de forma breve de qué se encargan cada una de las capas en un motor gráfico (Vallejo & Cleto, 2015) usando el esquema simplificado de la Figura 4:

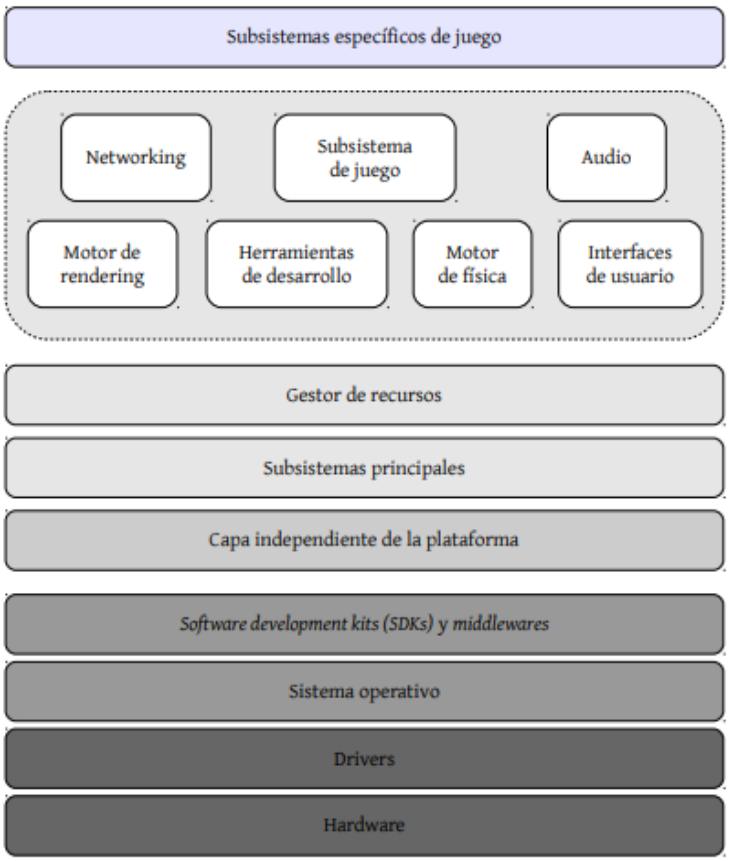

Figura 4: Arquitectura simplificada del motor gráfico. Fuente:(Vallejo & Cleto, 2015)

1. **Capa de hardware**

Esta capa se encuentra vinculada a la plataforma donde se ejecuta el motor de juego en cuestión, como una consola de sobremesa.

2. **Capa de drivers**

Esta capa contiene los componentes necesarios que permiten una gestión adecuada de los dispositivos implicados.

3. **Capa del sistema operativo**

Esta capa se encara de la comunicación de procesos que se ejecutan en el sistema operativo y el hardware vinculado a la plataforma en la que se está utilizando.

4. **Capa de SDKs y middlewares**

En esta capa se encuentran aquellas bibliotecas externas y SDK que, al igual que otros proyectos software, son utilizados por los desarrolladores adaptándolos a sus necesidades específicas, en este caso, en consolas de sobremesa y portátiles.

Un ejemplo de esto, es OpenGL y Direct3D, dos APIs muy famosas en este contexto, pero que mientras Direct3D se encuentra totalmente vinculado a los sistemas operativos de Windows, OpenGL es multiplataforma.

5. Capa independiente de la plataforma

Esta capa se encarga de aislar al resto de las superiores de cualquier aspecto que se encuentre vinculado a la plataforma en cuestión. De este modo, se pueden llevar a cabo el desarrollo de un videojuego para PC y una consola de sobremesa a la vez.

6. Capa de subsistemas

En esta capa se encuentran todas aquellas herramientas que completan el motor gráfico. Mientras que algunas son comunes a utilizar en cualquier proyecto software, otras son específicas para el desarrollo de videojuegos, como estructuras de datos y algoritmos, gestión de memoria, bibliotecas matemáticas, así como depuración y logging.

7. Capa de gestor de recursos

Esta capa se encarga de proporcionar al desarrollador una interfaz unificada para el acceso a los distintos módulos que componen nuestro motor gráfico, como los objetos 3D usados o recursos para la creación de la escena.

8. Motor de renderizado

El rendering o renderizado es una de las partes más importantes de cualquier juego, siendo esta una de las capas más complejas de un motor de videojuegos.

Este motor a su vez se divide en varias capas (Figura 5):

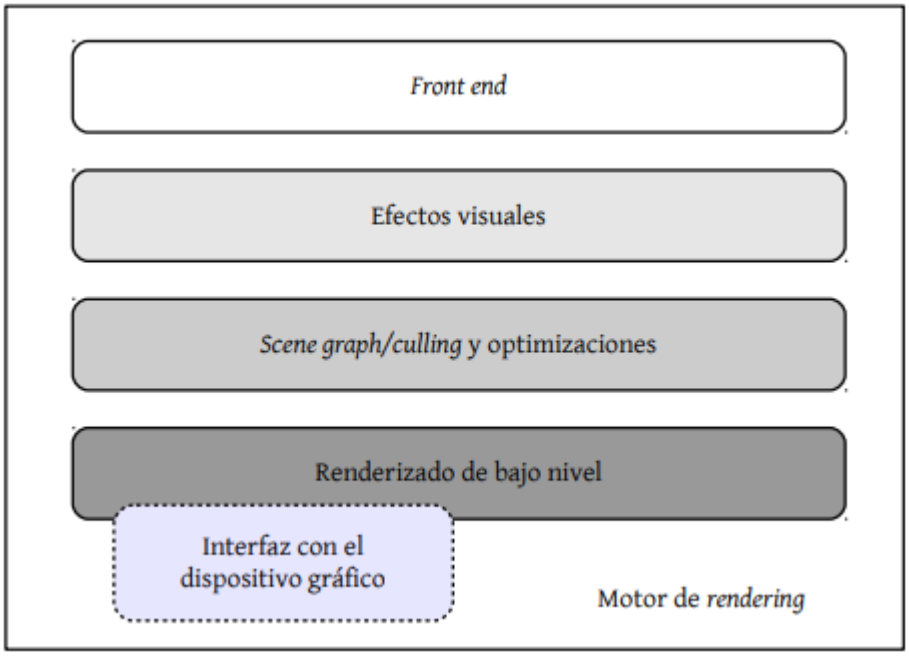

Figura 5: Esquema de motor de rendering. Fuente:(Vallejo & Cleto, 2015)

- **Renderizado a bajo nivel:** Capa que recoge las herramientas necesarias para la representación gráfica de las entidades de la escena como cámaras, objetos y materiales.
- **Scene graph/culling y optimizaciones:** Capa que se encarga de seleccionar qué parte de la escena se va a renderizar para mostrarse en pantalla. Limitando de esta forma la escena que se va a mostrar, se optimiza el rendimiento del motor de rendering.
- **Efectos visuales:** Capa que ofrece algunos efectos como sistemas de partículas y sombras dinámicas
- **Front-end:** Capa que se encarga de la superposición de elementos 2D en la escena como menús e interfaces gráficas como estado del jugador en un videojuego

9. Herramientas de depuración

Puesto que los videojuegos son aplicaciones en tiempo real, es necesario tener herramientas y utilidades que permitan al desarrollador depurar y optimizar el código para tener como resultado un mejor rendimiento del software desarrollado. Debido a esto un motor gráfico debe contar con herramientas para determinar tiempos de ejecución de cierto código, mostrar el rendimiento del motor mientras el juego está en ejecución, cuantificar la memoria utilizada y creación de logs, entre otras.

10. Motor de física

En los videojuegos, algo fundamental es la detección de colisiones, del cual se encarga el sistema de colisiones que se encarga de tres tareas fundamentales: detectar colisiones, determinar el punto donde se produce la colisión y realizar una respuesta tras la colisión.

Para esto, también se tiene en cuenta el hecho de que se intenta llegar a unas respuestas lo más realistas posibles, por lo que algunos juegos incluyen sistemas de físicas realistas o semirealistas que nos permiten esto. A estos sistemas se les denomina sistema o motor de física.

11. Interfaces de usuario

En la mayoría de proyectos software es necesario utilizar interfaces de usuario que se encarguen de procesar aquellos eventos de entrada que realiza el usuario. Esto no es diferente en el ámbito de los videojuegos, por lo que se cuenta también con una capa que permite procesar tanto los eventos de entrada como de salida, como puede serla vibración en un mando por una explosión producida.

12. Conexión a red o Networking

La inmensa mayoría de los juegos hoy en día cuenta con un modo online que permite jugar a dicho juego en compañía de otros jugadores convirtiéndose entonces en un juego multijugador.

Este módulo se encarga de comunicar cómo va cambiando el juego a los distintos usuarios que juegan enviando paquetes de información o sockets.

13. Subsistema de juego

En este bloque nos encontramos con todos los módulos que determinan el funcionamiento interno del videojuego, como propiedades del mundo y de los personajes. El subsistema de juego nos permite definir las reglas en las que se mueve el juego, es decir, mecánicas del personaje, objetivos a conseguir, etc.

En este bloque nos encontramos el que sería el sistema de eventos, el cual se encarga de comunicar los elementos y objetos. También nos encontramos por otro lado el sistema de scripting, donde se permite al desarrollador crear la lógica del juego.

Además de estos sistemas, en este bloque también podemos encontrar algunas utilidades para el tratamiento de la Inteligencia Artificial (IA).

14. Sistema de audio

Mientras que, al principio, el componente gráfico era el pilar central de un videojuego, hoy en día es igual de importante el audio que reproduce el juego y que acompaña a la acción visual. Debido a esto, es necesario contar con un motor de audio en el motor de videojuegos.

15. Capa de subsistemas específicos de juego

Esta se sitúa en la parte superior del esquema en el que se basa la arquitectura de un motor gráfico. En ella se encuentran aquellos módulos que ofrecen características propias del juego en cuestión. Aquí podemos encontrar algunos módulos como los encargados del sistema de cámaras, IA de los personajes no controlados por el usuario, sistemas de armas, etc.

3.2 Motores gráficos más utilizados

Si bien hemos podido ver en anteriores apartados que hay motores gráficos para todo tipo de usos, ahora nos centraremos en analizar de forma más detallada cuáles son los motores de videojuegos más utilizados y el porqué de esto.

3.2.1 Unity

Unity es el motor de juego más utilizado por los programadores de videojuegos debido a que su uso es muy sencillo junto a una gran potencia.

Con este motor se puede desarrollar cualquier juego, teniendo una cantidad de contenido muy grande y diversa, no solo en su motor, sino también en su tienda de assets que agregan una funcionalidad casi infinita. Además, cuenta también con una gran comunidad que se ha creado alrededor de este motor y una gran cantidad de cursos certificados gratuitos.

Unity 5 (*Unity Real-Time Development Platform | 3D, 2D VR & AR Engine*, s. f.) es la última versión de este motor y se puede descargar de forma gratuita siempre y cuando no se exceda del límite de ingresos de 100.000 dólares al año. Esto es otro de los atractivos de Unity, ya que tiene mucha flexibilidad a la hora de adaptarse a las necesidades de cada desarrollador, teniendo diversos planes con funcionalidades diferentes según convenga.

Otro de sus atractivos es la cantidad de formatos que soporta a la hora de exportar los videojuegos creados, pudiendo exportar dichos juegos a plataformas móviles, como Android o iOS, plataformas y consolas de sobremesa, como Xbox y PlayStation, y, por supuesto, a PC.

Algunos juegos muy famosos creados con este motor gráfico son Lara Croft Go y Pillars of Eternity.

3.2.2 Unreal Engine

Como se ha comentado en un apartado anterior, Unreal Engine (*Unreal Engine | The Most Powerful Real-Time 3D Creation Tool*, s. f.) es un motor que lleva mucho tiempo en la historia de los motores gráficos, siendo lanzada su primera versión en 1998.

Este motor sigue siendo uno de los más populares y se sigue utilizando a día de hoy trayendo títulos tan importantes como Gears of War o Mass Effect.

En 2021 se lanzó su última versión, Unreal Engine 5, creando un gran revuelo por sus gráficos e iluminación mejorada con RayTracing. Esta última versión sigue siendo gratuita, pero teniendo en cuenta los términos y condiciones que Epic Games tiene para el desarrollo en este motor gráfico.

Como Unity, se puede llegar a exportar a una gran variedad de plataformas.

3.2.3 Game Maker Studio

Este motor tiene como última versión GameMaker Studio 2 (*GameMaker*, s. f.), un motor gráfico de pago, pero que permite a los usuarios una prueba gratuita antes de realizar un pago de unos 5$ mensuales.

La programación en este motor puede resultar un poco más complicada, puesto que se debe programar de acuerdo a su propio lenguaje de programación llamado GML (Game Maker Language). Sin embargo, este lenguaje resulta muy similar a los lenguajes comúnmente usados para el desarrollo de videojuegos, como C++ o C#.

Este motor es el más usado por aquellos desarrolladores que quieren centrarse en el desarrollo de juegos con gráficos en 2D, pero este motor realmente no se encuentra limitado por este tipo de gráficos, ya que su editor de assets y mapas permite jugar con la perspectiva para el desarrollo de un videojuego con una tercera dimensión.

Algunos de los juegos más conocidos que se han creado con este motor es Undertale, Spelunky y Hotline Miami, entre otros.

3.2.4 RPG Maker

RPG Maker (*Make Your Own Game with RPG Maker*, s. f.) ha sido y seguirá siendo el motor de iniciación por parte de aquellos usuarios que no tienen conocimientos de desarrollo en videojuegos, pero que aun así disfrutan creando su propio juego con gráficos sencillos.

Este motor se especializa en la creación de juegos RPG al estilo de la saga Final Fantasy en sus primeras entregas, teniendo unos gráficos pixel art en vista isométrica. Este motor incluye un editor de mapas, editor de bases de datos, etc.

Para el uso de este motor es necesario conocimiento en C++ o Java, utilizando este último con el objetivo de desarrollar juegos para plataformas móviles o HTML5.

Algunos juegos famosos creados con este modesto motor son Tot he moon y LISA: The First.

3.2.5 Otros motores

Además de todos los motores mencionados anteriormente, hay muchas más opciones en el mercado que también tienen algunas características muy atractivas para el desarrollo de videojuegos. A continuación, se muestran de forma breve algunos motores menos conocidos, pero que también pueden ser interesantes:

a. CryEngine

Motor gráfico nacido de la empresa Crytek y usado para el desarrollo del famoso juego Far Cry (*CRYENGINE | The Complete Solution for next Generation Game Development by Crytek*, s. f.).

b. Godot

Motor de software libre y código abierto que puede modificarse según conveniencia (*Godot Engine - Free and Open Source 2D and 3D Game Engine*, s. f.). Es totalmente gratuito y soporta lenguajes de programación como C#, C++, Visual Scripting y GDSScript (lenguaje específico de este motor).

c. Stencyl

Herramienta de creación de juegos en 2D, que permite al usuario desarrollar un juego sin tener conocimientos de programación mediante un sistema de "arrastrar y soltar" (*Stencyl: Make iPhone, iPad, Android & Flash Games without code*, s. f.).

d. Source

Motor gráfico de la empresa Valve (*Valve Corporation*, s. f.), la cual también es responsable de la plataforma de videojuegos Steam. Fácil de manejar para personas con menos experiencia y especializado en la creación de videojuegos en primera persona.

e. Microsoft XNA

Motor basado en lenguajes de programación .NET (*Microsoft XNA Framework Redistributable 4.0*, s. f.). Cuenta con la mayoría de las librerías de .NET, así como algunas de las librerías específicas XNA. Permite la creación de aplicaciones para PC y Xbox 360.

f. Torque 3D

Nacido en 2012 a partir de su predecesor Torque Game Engine, Torque 3D (*Torque3D*, s. f.) fue desarrollado por Dynamix para el videojuego Tribes 2. Está especializado en la creación de videojuegos de disparos en primera persona (FPS o First-Person Shooter).

g. Pico-8

Pico-8 es una plataforma en la que el usuario puede, además de jugar, crear y compartir videojuegos modestos con una estética pixel art (*PICO-8 Fantasy Console*, s. f.). Este motor es muy fácil de usar por aquellos usuarios que no hayan usado nunca un motor gráfico. Se especializa en juegos 2D y requiere de un pequeño pago de 15€ para poder utilizarlo.

3.2.6 Tabla de motores gráficos y fuentes

A continuación, se muestran los motores vistos en este estudio epistemológico junto con sus respectivas páginas web en la Tabla 1:

Motor gráfico	Características	Página web
Unity	Fácil de aprender. Licencia gratuita muy amplia. Gran comunidad y contenido.	https://unity.com/
UNREAL ENGINE	Licencia gratuita más limitada. Gráficos muy realistas. Comunidad algo más pequeña.	https://www.unrealengine.com/
GameMaker	Especializado en 2D. Programación más avanzada. Opción de programar con el método “arrastrar y soltar”. Licencia gratuita muy limitada.	https://gamemaker.io/es/gamemaker
RPG MAKER	No es necesario saber de programación. Gran comunidad y contenido.	https://www.rpgmakerweb.com/
CRYENGINE	Aprendizaje algo complicado. Gran edición de terrenos. Licencia gratuita más limitada.	https://www.cryengine.com/
GODOT Game engine	Licencia gratuita. Mucha documentación. Soporta menos plataformas. Menos recursos disponibles.	https://godotengine.org/
stencyl	Fácil de aprender. Incluye programación con Scratch. Gran cantidad de documentación. Solo para juegos 2D.	https://www.stencyl.com/
source	Fácil de aprender. No es gratuito. Algunos bugs no corregidos.	https://www.valvesoftware.com/es/
xna	Entorno y SDKs gratuitos. Soporta menos plataformas. Soporta programación directa sobre GPU.	https://www.microsoft.com/en-us/download/details.aspx?id=23714
TORQUE	Gran trabajo de iluminación y sombreado. Totalmente gratuito con licencia MIT.	https://torque3d.org/
PICO-8 FANTASY CONSOLE	Solo soporta hasta 6 botones en juegos. Especializado en juegos retro. Fácil de programar.	https://www.lexaloffle.com/pico-8.php

Tabla 1: Resumen de los motores gráficos estudiados

3.3 Uso del motor gráfico

Habiendo visto en el apartado anterior cuales son los motores de videojuegos más usados, nos centraremos en uno de ellos para explicar más a fondo cómo se usaría uno de estos motores, concretamente, Unity.

Veremos de forma breve qué funcionalidades pone este motor a disposición de un desarrollador y cómo usarlas. Esto se hará con la creación de un prototipo de videojuego que muestre algunas de estas utilidades y funciones.

3.3.1 Creación el proyecto

Haciendo uso de Unity, se comienza con la creación de un proyecto en blanco, teniendo que indicar qué tipo de juego será y su nombre. Para este caso podemos elegir un proyecto en blanco en 3D (Figura 6).

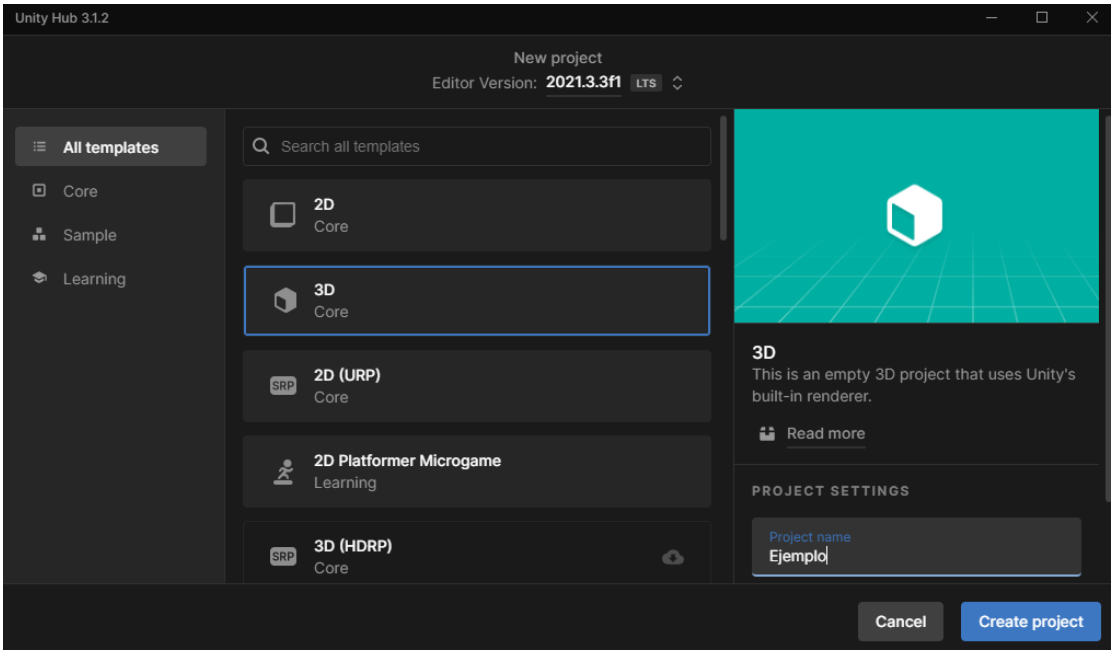

Figura 6: Creación de un proyecto en Unity. Fuente: Elaboración propia

Tras la creación de este, ya se podría empezar a crear la escena de un nivel de videojuego en el editor de Unity.

3.3.2 Creación de la escena

En nuestro ejemplo, vamos a ver cómo el jugador puede ir saltando por unas plataformas en 3D para llegar a un sitio más alto.

Para hacer esto, se deben de crear GameObjects en la escena que se correspondan con las plataformas correspondientes. Además, una vez creados se deben de modificar sus “Transforms”, es decir, la propiedad que guarda la posición, rotación y escalado de los GameObjects (estos son los objetos que utiliza Unity).

Una vez creada la escena (Figura 7), podemos también crear nuevos materiales y asociarlos a cada GameObject, de forma que su aspecto se vea modificado, añadir luces de distintos tipos y añadir texturas si lo deseamos.

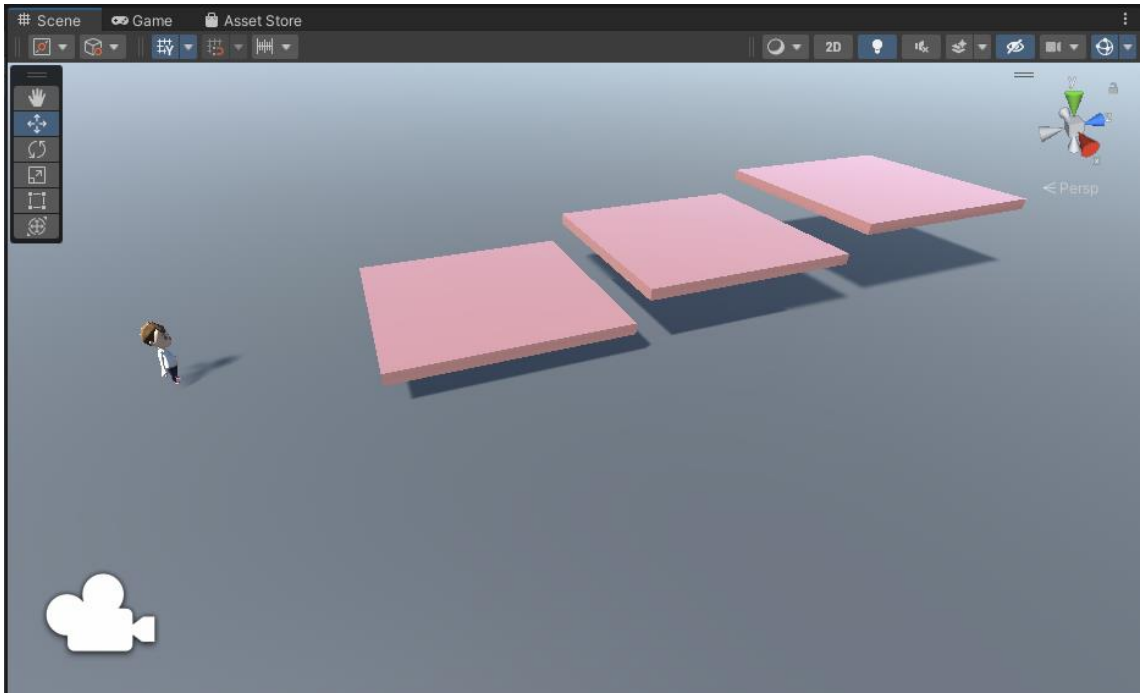

Figura 7: Escena del juego. Fuente: Elaboración propia

Además de esto, se pueden ajustar otros elementos necesarios como puede ser la cámara o añadir un tipo de luz determinado.

En el caso del ejemplo, se ha colocado una luz direccional, que como se ve en la Figura 7, se intuye que esta se coloca en la parte superior izquierda. Además, también se ha colocado la cámara en un ángulo en el que el personaje se ve perfectamente para poder controlarlo correctamente.

3.3.3 Descarga de Assets

Una de las grandes ventajas del uso de Unity es la gran cantidad de Assets disponibles que se encuentran en la tienda (Figura 8). Muchos de ellos son gratuitos, por lo que resulta muy interesante echar un vistazo a estos para crear un videojuego antes de crear unos propios.

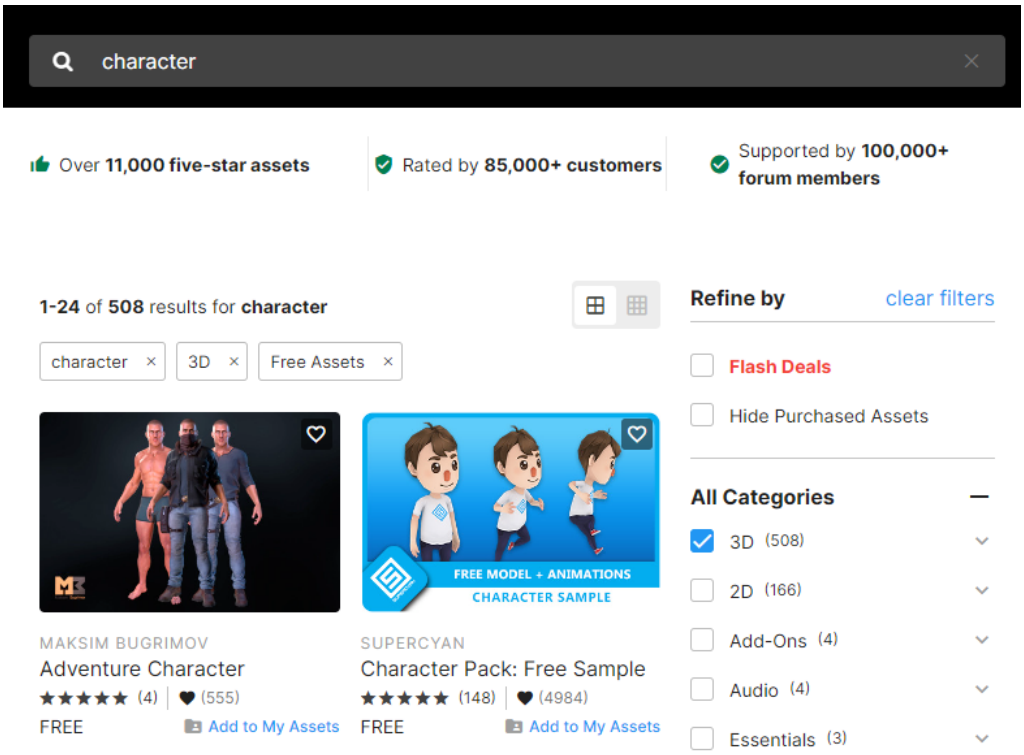

Figura 8: Tienda de Assets de Unity. Fuente: (Unity Asset Store, s. f.)

Para el ejemplo del prototipo se podrían descargar los Assets necesarios para añadir el personaje que controlará el jugador.

3.3.4 Scripting

Es necesario dar lógica al juego que se quiere crear y para ello se deben crear ciertos scripts. Unity permite crear estos scripts en C# usando Visual Studio o en su propio lenguaje de programación, UnityScript.

Gracias a estos scripts, el desarrollador puede controlar las acciones del jugador, mover la cámara correctamente, añadir movimiento y acciones a los enemigos, crear objetos de forma automatizada, etc.

En el caso del ejemplo creado, se podrían ver los scripts que controlan al personaje o la cámara en el editor de Unity o bien abriendo estos en la herramienta de Visual Studio.

3.3.5 Animaciones

Unity permite gestionar animaciones de una forma muy sencilla gracias a su herramienta “Animator”. De esta forma, el desarrollador puede determinar cuándo se hace una animación u otra creando un diagrama que permite hacer esto de forma mucho más visual.

En el caso del personaje que se ha descargado para el prototipo, se puede ver este diagrama y cambiar sus animaciones en el editor de Unity (Figura 9).

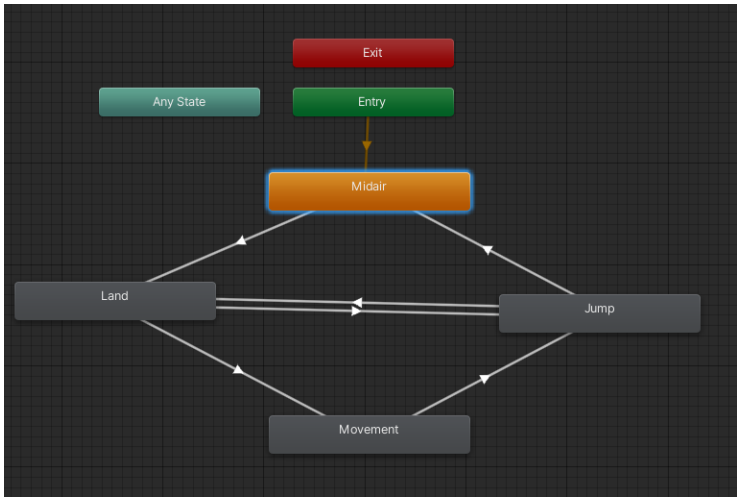

Figura 9: Animator de Unity. Fuente: Elaboración propia

3.3.1 Distribución del proyecto

Una vez que está terminado el proyecto, Unity permite la distribución en distintas plataformas digitales (Figura 10), pudiendo elegir su compilación para plataformas desde PC y Linux, pasando por plataformas móviles, hasta consolas de sobremesa como Xbox, PlayStation, etc.

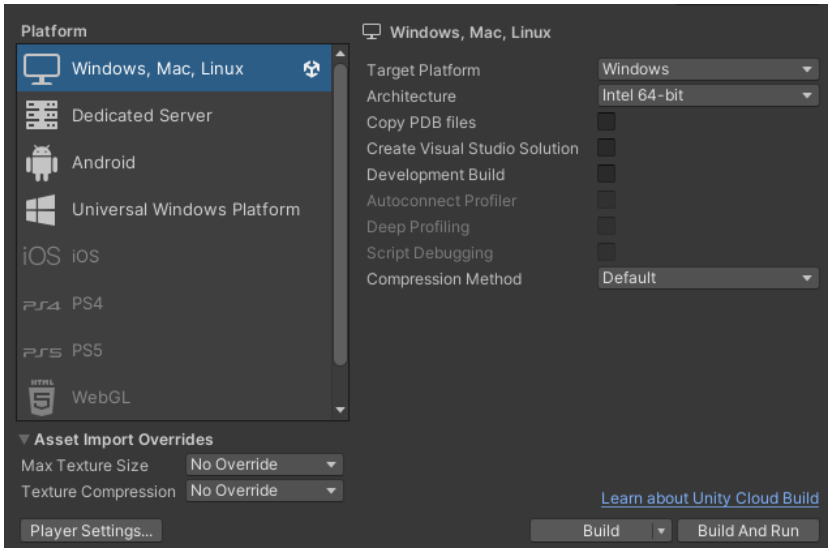

Figura 10: Compilar proyecto en Unity. Fuente: Elaboración propia

4. Tendencias futuras

Como se ha podido observar a lo largo de este trabajo, la evolución en el ámbito de los motores gráficos y, por consiguiente, en los videojuegos, se ha dado de forma muy rápida y a un ritmo que muy pocos podrían imaginar.

Cada vez más, la industria del videojuego se hace más importante en el panorama de la economía mundial, siendo este sector uno de los principales encargados de dar a la población una buena parte del tiempo de ocio, más aún si tenemos en cuenta que las generaciones venideras se encontrarán acompañadas de la tecnología desde sus primeros días.

La creación de videojuegos intenta poner mucha más atención al jugador, siendo este el pilar central dentro del universo del juego. Esto es, el jugador debe ser el personaje principal y debe sentirse protagonista de la trama del juego.

Además, la aplicación de videojuegos en otros ámbitos como el trabajo, la salud o la educación, también da la oportunidad de expandir las áreas de trabajo que se investigan.

A continuación, veremos algunas de las tendencias en el uso y desarrollo de los motores de videojuegos.

4.1 Mejora general de gráficos

Por supuesto, una rama del estudio de los motores de videojuegos es el apartado de mejoras gráficas, pero no se pueden obviar los avances en este aspecto.

Ya fue espectacular en su momento la presentación del motor gráfico Unreal Engine 5, donde podíamos ver unos gráficos impresionantes acompañados de una iluminación muy realista gracias al Ray Tracing. Sin embargo, esto solo es una muestra de lo que se podría llegar a hacer.

Desde la presentación de la nueva versión del motor gráfico de Unreal se han hecho aún más mejoras, llegando a unos niveles de realismo espectaculares. Esto se puede ver fácilmente en la Figura 11, imagen de la presentación que se hizo recientemente de una demo que recreaba una estación japones a un nivel de detalle que parece casi imposible saber si era real o no (Arroyo, 2022).

Figura 11: Recreación de una estación japonesa en Unreal Engine 5. Fuente: (Arroyo, 2022)

En este aspecto, los paisajes y escenarios pueden ser algo "fácil", pero donde realmente impactan estas mejoras es en la creación de modelos humanos lo más realistas posible.

Este es el punto donde más se está trabajando, y es que estos "humanos" serán clave para la expansión de los motores gráficos y, adelantándonos un poco a un apartado posterior, la creación de un metaverso. Un ejemplo de la mejora en la creación de rostros lo vemos en Enemies (Raya, 2022), una cinemática realmente sorprendente creada con el motor gráfico Unity (Figura 12).

Figura 12: Cinemática "Enemies" en Unity. Fuente: (Raya, 2022)

Las mejoras gráficas de estos motores están abriendo la posibilidad a la inclusión de estos como herramienta creativa en la industria cinematográfica. Esto lo vemos en una serie de televisión, El mandaloriano, donde se hizo uso del motor gráfico Stagecraft, en lugar de la ya conocida pantalla verde (Villalobos, 2022).

Esto abre la posibilidad de la búsqueda de nuevos perfiles profesionales, donde los equipos de la industria del cine contarían con especialistas en motores gráficos, artistas 3D con conocimientos tanto de cine como de videojuegos, especialistas de fotografía que suman a su conocimiento el saber acerca de los motores de iluminación, etc.

4.2 IA para la creación de gráficos

Seguimos en la mejora de los gráficos en los motores de videojuegos, pero con una especial mención a la IA

Si bien la IA se ha usado hasta ahora para controlar personajes en un videojuego o recrear conversaciones realistas en un lenguaje que nos parezca lo más natural posible en nuestros asistentes móviles, la IA puede abarcar mucho más, siendo uno de estos campos la generación de gráficos.

Un ejemplo de este campo de estudio lo tenemos en GauGAN2 (Figura 13), una herramienta desarrollada por Nvidia que permite la creación de gráficos realistas a partir de dibujos muy simples que el usuario puede hacer (Arsuaga, 2022).

Figura 13: Ejemplo de GauGAN. Fuente: (Arsuaga, 2022)

GAN o Generative Adversarial Network es un sistema que usa dos redes neuronales que se enfrentan la una a la otra para crear este tipo de imágenes. Mientras que la primera red neuronal se encarga de generar las imágenes, la segunda se enfrenta a esta intentando detectar si la imagen es real o es falsa/creada, lo que al final resulta en unas imágenes que han pasado un filtro muy estricto y de ahí su gran fidelidad a la realidad.

Dando un paso más allá, nos encontramos con el concepto de “Superresolución” y, de nuevo, con la tecnología de Nvidia.

Este concepto de “Superresolución” se resume en la capacidad de generar imágenes a baja resolución para después reescalarlas usando redes neuronales GAN. Esto contribuye a la fluidez que podemos ver en la imagen de, por ejemplo, un videojuego jugado en 4k.

Nvidia dispone de una parte específica de sus GPUs llamadas TPU cores. Estas TPU están especializadas en ejecutar este tipo de algoritmos de superresolución y a esta tecnología la empresa la ha llamado Deep Learning Super Sampling (DLSS) (NVIDIA DLSS, s. f.).

4.3 La realidad virtual (RV) y el metaverso

Al igual que con la realidad aumentada, este campo resulta ser uno de los más investigados en cuanto a la creación de escenas y entornos en 3D, teniendo que adaptarse los motores gráficos a este nuevo escenario.

Actualmente nos encontramos en un punto en el que la realidad virtual tiene mucho que decir, y es que, si bien llevamos ya años trabajando en la realidad virtual, Mark Zuckerberg, creador de la gran red social de Facebook, anunció el año pasado que ya trabaja en la creación y desarrollo de lo que será la guinda del pastel, el llamado “metaverso” (El Economista, 2021).

Este concepto es aún muy abierto, pero podemos definirlo como un nuevo mundo, paralelo a la vida misma, que se desarrolla de forma virtual en internet y al que tendremos acceso en un futuro gracias a diversos dispositivos que permitan una inmersión total del usuario. Hablamos de una experiencia parecida a la que nos muestran algunas películas como Ready: Player One con herramientas y dispositivos que ayuden a conseguir una experiencia lo más inmersiva posible (Figura 14Figura 14).

Figura 14: Dispositivos de Realidad Virtual. Fuente:(El Economista, 2021)

5. Conclusiones

Una vez realizado este estudio, se puede observar que los campos de aplicación y las consecuencias derivadas del desarrollo en el ámbito de los sistemas y motores gráficos suponen un gran impacto a día de hoy.

Si bien los videojuegos son los más beneficiados a la hora de esta evolución de los motores gráficos, estas mejoras se aplican a muchos más sectores, como la industria del cine.

La enseñanza no es algo que quede fuera del alcance de esta evolución, pues ya se puede ver en algunas aulas cómo los profesores enseñan a sus alumnos con nuevos recursos como la realidad aumentada o la realidad virtual en asignaturas en las que pueden llegar a ser de gran ayuda: contemplar una figura en realidad aumentada para conseguir entrenar la visión espacial y sacar las vistas correctas en dibujo técnico, contemplar en tiempo real medidas de nuestro entorno y posibles relaciones geométricas, visitar de forma virtual algún museo y poder ver de cerca un cuadro en el que se aprecien de forma fiel las pinceladas del autor, etc.

Las aplicaciones de los gráficos por ordenador y todas sus ramas de estudio son muchas y, para ello, no solo hace falta saber manejar un motor gráfico como Unity, sino que se necesitan conocimientos en campos como matemáticas, física, programación, incluso arte o fotografía para modelado e iluminación. Dicho esto, resulta realmente interesante observar cómo los gráficos por ordenador es una de las áreas más exigentes dentro de las ciencias de la computación.

Si algo está claro es que los sistemas y motores gráficos han supuesto un gran avance en la computación y en la forma de vivir el día a día de muchas personas.

Proyección didáctica

En este bloque se expondrá todo lo relativo a la unidad didáctica elaborada para el tema “Motores Gráficos” en el que se centra este Trabajo de Fin de Máster.

La unidad didáctica que se va a desarrollar versa sobre el tema “Desarrollo de videojuegos para plataformas móviles” que se encontraría dentro del módulo profesional de “Programación multimedia y dispositivos móviles” el cual pertenece al Ciclo Formativo de Grado Superior de Desarrollo de Aplicaciones Multiplataforma.

Cabe destacar que esta unidad, al encontrarse dentro del módulo de “Programación multimedia y dispositivos móviles” solo la podrán impartir aquellas personas pertenecientes al cuerpo de Profesores de Enseñanza Secundaria (PES).

Para comenzar, en este bloque se explicarán las características del centro donde se impartiría dicha unidad didáctica (6. Características del centro), indicando la ubicación del mismo, el alumnado objetivo, así como los materiales e instalaciones disponibles en el mismo.

Tras esto, comenzará la explicación en profundidad de la unidad didáctica desarrollada de acuerdo con las normativas vigentes, explicando también la metodología aplicada y los criterios de evaluación que se han estimado oportunos.

Posteriormente, se desarrolla un breve apartado donde se tratará de forma específica todo lo relacionado con la atención a la diversidad (8. Atención a la diversidad).

Por último, se exponen las conclusiones obtenidas (9. Conclusiones) como resultado del proceso de la creación de la unidad didáctica dispuesta en este trabajo.

6. Características del centro

Esta unidad didáctica, incluida en el Ciclo Formativo de Grado Superior de Desarrollo de Aplicaciones Multiplataforma, se impartirá en un centro que dispone de este ciclo formativo. El nombre del instituto en cuestión es “IES ejemplo” cuyo correo institucional es *centroejemplo@iesejemplo.es*.

Este centro cuenta con 136 profesores contando todos los niveles educativos que se ofertan en él.

6.1 Ubicación del cetro

El centro “IES ejemplo” es un centro ficticio que se encontraría situado en Jaén capital, ciudad que cuenta con un número aproximado de 111 932 habitantes. En concreto, el centro se encuentra en la zona oeste de la ciudad.

En la Figura 15 puede verse la situación exacta del centro.

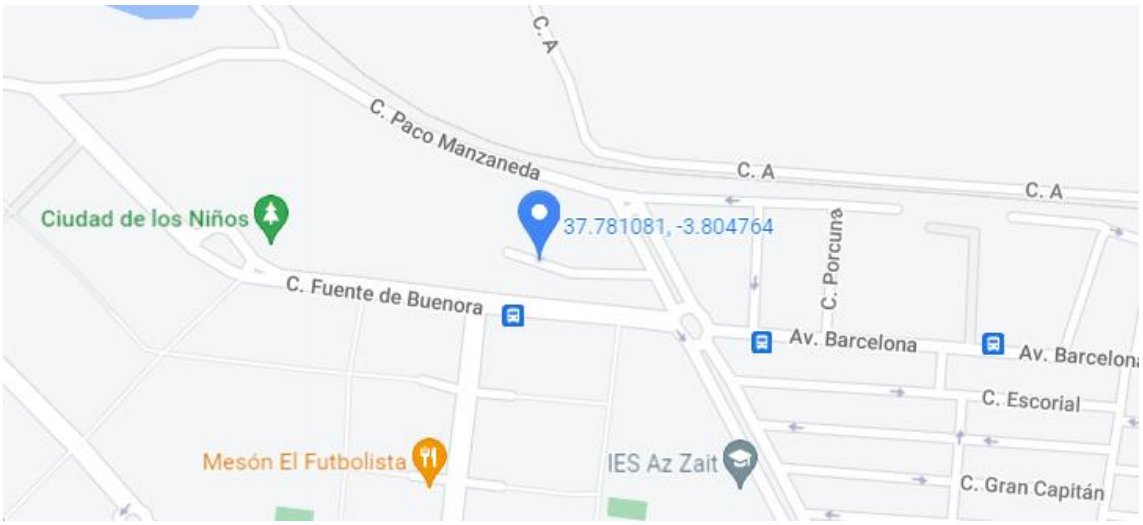

Figura 15: Situación del centro. Fuente: Elaboración propia

6.2 Niveles de enseñanza ofertados

El centro cuenta con las siguientes enseñanzas:

- Educación Secundaria Obligatoria
- Bachillerato
- Formación Profesional Básica
- Formación Profesional Básica Específica
- Ciclos Formativos de Grado Medio
- Ciclos Formativos de Grado Superior

Además, el centro implantó años atrás la enseñanza bilingüe en todos los niveles de enseñanza del centro.

Entre los ciclos que se imparten se encuentra el ciclo ya mencionado donde se enmarca esta UD, así como el Ciclo Formativo Superior de Sistemas Informáticos en Red, el Ciclo Formativo de Grado Superior de Desarrollo de Aplicaciones Web y el Ciclo Formativo de Grado Medio en Sistemas Microinformáticos y Redes.

6.3 Contexto del alumnado y entorno socio-económico

El centro “IES ejemplo” se sitúa en medio de varios barrios de donde procede fundamentalmente la mayoría del alumnado de Educación Secundaria Obligatoria y Bachillerato.

En el entorno del centro se encuentran cinco colegios públicos de educación infantil y primaria y tres institutos de secundaria desde los que procede una gran parte del alumnado cuando promocionan. Estos alumnos cuentan con un nivel socio-económico de clase media-alta. Por los barrios que rodean al centro, algunos alumnos tienen procedencia extranjera, lo que conlleva una diversidad de culturas en el centro. Aun así, estos alumnos son una minoría, con lo que pueden llegar a presentar problemas a la hora de integrarse en el centro.

Según se ha observado, los alumnos suelen tener actividades extraescolares como conservatorio, escuelas de idiomas u otras actividades como deportes o talleres. Además, atendiendo a la naturaleza tecnológica del módulo donde se imparte la unidad didáctica, cabe destacar que los alumnos disponen todos ellos de, al menos, un ordenador con conexión a Internet en sus casas.

Como norma general, las familias de los alumnos que ingresan en el centro son familias con un buen grado de estabilidad, donde no se encuentran grandes problemas que puedan condicionar la vida en el centro del estudiante. Además, presentan un gran nivel de interés en el desarrollo del estudiante a nivel académico.

El alumnado al que se impartirá clase constará de 20 alumnos en el aula (12 hombres y 8 mujeres). Proviene en su mayoría del grado medio de Técnico en Sistemas Microinformáticos y Redes (12 alumnos). 5 de los alumnos provienen de Bachillerato, mientras que 3 de ellos se han cambiado de otros ciclos superiores (1 de ASIR y 2 de DAW).

Además de esto, cabe resaltar que uno de los alumnos tiene discapacidad motora, teniendo que desplazarse en silla de ruedas. Por ello, se le tendrá que facilitar el acceso al aula y a su puesto de trabajo.

6.4 Instalaciones y materiales del centro

El centro “IES ejemplo” cuenta con una superficie útil de 9500 metros cuadrados, y un total de 5 edificios, donde se imparten las clases correspondientes a la ESO, Bachillerato y Ciclos Formativos.

Cuentan con una extensa zona de recreo, donde se incluye también una pista donde se puede realizar la práctica de varios deportes y que es aprovechada también para impartir clases en algunos ciclos relacionados con el deporte.

Todas las aulas del centro disponen de proyectores, un ordenador en el puesto del profesor y una pizarra blanca con rotuladores. Desde todas ellas, el profesorado puede conectarse a Internet por cable con el ordenador que dispone el profesor en cada aula, así como por conexión wifi, disponible en todos los edificios del centro.

Como se puede ver, el centro cuenta con una buena integración de las TIC, contando también con aulas especializadas para los diferentes ciclos formativos, entre las que se encuentran varias aulas o laboratorios de informática. Estas aulas cuentan con enchufes en cada puesto de trabajo, donde el alumnado puede enchufar su ordenador personal. El centro también dispone de ordenadores portátiles que se darán a los alumnos en caso de que no tengan disponible un ordenador propio.

Además de esto, cabe destacar que el centro también dispone de la red Moodle andaluza en la que los profesores pueden subir todo aquello relacionado con las asignaturas y módulos que imparten.

7. Unidad Didáctica

En este apartado se desarrollará la unidad didáctica que se pretende impartir en el centro anteriormente descrito. Se tendrá en cuenta la normativa vigente para todo lo referente a competencias generales, objetivos, contenidos, criterios de evaluación, etc.

También se planificarán las actividades correspondientes a esta unidad didáctica y su metodología (7.11. Metodología) y se resumirá todo de forma mucho más visual en una tabla que contendrá toda la información relevante de dicha unidad didáctica (7.13. Tabla resumen de la UD).

7.1 Normativa

Esta unidad didáctica se ha creado teniendo en cuenta la siguiente normativa:

- **Real Decreto 450/2010**, de 16 de abril, por el que se establece el título de Técnico Superior en Desarrollo de Aplicaciones Multiplataforma y se fijan sus enseñanzas mínimas.
- **Orden de 16 de junio de 2011**, por la que se desarrolla el currículo correspondiente al título de Técnico Superior en Desarrollo de Aplicaciones Multiplataforma.

7.2 Competencia general

La competencia general de este título consiste en desarrollar, implantar, documentar y mantener aplicaciones informáticas multiplataforma, utilizando tecnologías y entornos de desarrollo específicos, garantizando el acceso a los datos de

forma segura y cumpliendo los criterios de «usabilidad» y calidad exigidas en los estándares establecidos.

7.3 Distribución temporal

Las sesiones son de 2 horas habiendo 2 sesiones semanales a lo largo de 4 semanas de clase en la segunda mitad del 2º trimestre (8 sesiones/16 horas en total para esta UD). Los días en los que se impartirán clases serán miércoles y viernes. Se tendrá en cuenta que esta es la última unidad didáctica antes de la Formación en Centros Profesionales o FCT (prácticas en empresas).

7.4 Objetivos generales (OG)

De los objetivos generales que se recogen en la **Orden de 16 de junio de 2011**, por la que se desarrolla el currículo correspondiente al título de Técnico Superior en Desarrollo de Aplicaciones Multiplataforma relacionadas con el módulo de "Programación multimedia y dispositivos móviles", **la unidad didáctica creada** contribuye a alcanzar **los objetivos generales:**

h) Emplear herramientas de desarrollo, lenguajes y componentes visuales, siguiendo las especificaciones y verificando interactividad y usabilidad, para desarrollar interfaces gráficos de usuario en aplicaciones multiplataforma.

i) Seleccionar y emplear técnicas, motores y entornos de desarrollo, evaluando sus posibilidades, para participar en el desarrollo de juegos y aplicaciones en el ámbito del entretenimiento.

j) Seleccionar y emplear técnicas, lenguajes y entornos de desarrollo, evaluando sus posibilidades, para desarrollar aplicaciones en teléfonos, PDA y otros dispositivos móviles.

s) Establecer procedimientos, verificando su funcionalidad, para desplegar y distribuir aplicaciones.

7.5 Competencias profesionales, personales y sociales (CPPS)

De las CPPS que se recogen en la **Orden de 16 de junio de 2011**, por la que se desarrolla el currículo correspondiente al título de Técnico Superior en Desarrollo de Aplicaciones Multiplataforma relacionadas con el módulo de "Programación multimedia y dispositivos móviles", la **unidad didáctica** desarrollada contemplará las siguientes:

d) Gestionar entornos de desarrollo adaptando su configuración en cada caso para permitir el desarrollo y despliegue de aplicaciones.

g) Integrar contenidos gráficos y componentes multimedia en aplicaciones multiplataforma, empleando herramientas específicas y cumpliendo los requerimientos establecidos.

h) Desarrollar interfaces gráficos de usuario interactivos y con la usabilidad adecuada, empleando componentes visuales estándar o implementando componentes visuales específicos.

i) Participar en el desarrollo de juegos y aplicaciones en el ámbito del entretenimiento y la educación empleando técnicas, motores y entornos de desarrollo específicos.

j) Desarrollar aplicaciones para teléfonos, PDA y otros dispositivos móviles empleando técnicas y entornos de desarrollo específicos.

m) Empaquetar aplicaciones para su distribución preparando paquetes auto instalables con asistentes incorporados.

t) Establecer vías eficaces de relación profesional y comunicación con sus superiores, compañeros y subordinados, respetando la autonomía y competencias de las distintas personas.

7.6 Orientaciones pedagógicas (OP)

Este módulo profesional contiene la formación necesaria para desempeñar la función de desarrollo de aplicaciones multimedia, juegos y aplicaciones adaptadas para su explotación en dispositivos móviles.

La función de desarrollo de aplicaciones multimedia, juegos y aplicaciones adaptadas para su explotación en dispositivos móviles incluye aspectos como:

1. La creación de aplicaciones que incluyen contenidos multimedia basadas en la inclusión de librerías específicas en función de la tecnología utilizada.
2. La creación de aplicaciones para dispositivos móviles que garantizan la persistencia de los datos y establecen conexiones para permitir su intercambio.
3. El desarrollo de juegos 2D y 3D utilizando las funcionalidades que ofrecen los motores de juegos, así como su puesta a punto e implantación en dispositivos móviles.

Las actividades profesionales asociadas a esta función se aplican en el desarrollo de software multiplataforma en empresas especializadas en la elaboración de contenidos multimedia, software de entretenimiento y juegos.

De las anteriores orientaciones pedagógicas, la **unidad didáctica** elaborada contempla la **número 3**. Esta orientación pedagógica se contemplará a la hora de realizar las actividades programadas que tienen como objetivo el desarrollo de videojuegos en plataformas móviles utilizando Unity como motor de videojuegos principal.

7.7 Líneas de actuación

De las líneas de actuación que se recogen en la **Orden de 16 de junio de 2011**, por la que se desarrolla el currículo correspondiente al título de Técnico Superior en Desarrollo de Aplicaciones Multiplataforma relacionadas con el módulo de "Programación multimedia y dispositivos móviles", **la unidad didáctica creada** contempla las siguientes:

1. El desarrollo de aplicaciones que integran objetos multimedia.
2. El análisis de motores de juegos, sus características y funcionalidades.
3. El desarrollo de juegos 2D y 3D aplicando técnicas específicas y utilizando instrucciones gráficas para establecer efectos sobre objetos o imágenes.

7.8 Contenidos

7.8.1 Contenidos básicos

Los contenidos básicos asociados a este módulo se dividen en los siguientes cuatro bloques:

1. Análisis de tecnologías para aplicaciones en dispositivos móviles.
2. Programación de aplicaciones para dispositivos móviles.
3. Utilización de librerías multimedia integradas.
4. Desarrollo de juegos 2D y 3D.

En el caso de la **unidad didáctica** que se pretende desarrollar, los contenidos que se van a impartir **corresponden a los del bloque 4, Desarrollo de juegos 2D y 3D,** siendo estos:

a) Entornos de desarrollo para juegos.

b) Integración del motor de juegos en entornos de desarrollo.

c) Conceptos avanzados de programación 3D.

d) Fases de desarrollo.

e) Propiedades de los objetos, luz, texturas, reflejos, sombras.

f) Aplicación de las funciones del motor gráfico. Renderización.

g) Aplicación de las funciones del grafo de escena. Tipos de nodos y su utilización.

h) Análisis de ejecución. Optimización del código.

7.8.2 Elementos transversales y educación en valores

En la explicación y desarrollo de los contenidos de esta UD, se incluirán también los siguientes temas trasversales:

- Igualdad de género.
- Energías renovables.

7.9 Objetivos específicos

Se han determinado los siguientes objetivos específicos que deberá cumplir el alumno para superar la unidad didáctica:

a. Conocer el motor de videojuegos Unity

b. Integrar el uso de Unity en entornos de desarrollo

c. Utilizar conceptos avanzados de programación 2D y 3D

d. Conocer las fases de desarrollo de un videojuego

e. Conocer y manejar propiedades de los objetos, luz, texturas, reflejos y sombras.

f. Conocer y aplicar las funciones del motor gráfico Unity.

g. Aplicar las funciones del grafo de escena.

h. Analizar, documentar y optimizar el código creado.

7.10 Resultados de aprendizaje y criterios de evaluación

Los resultados de aprendizaje que se asocian a este módulo son los siguientes:

1. Aplica tecnologías de desarrollo para dispositivos móviles evaluando sus características y capacidades.
2. Desarrolla aplicaciones para dispositivos móviles analizando y empleando las tecnologías y librerías específicas.

3. Desarrolla programas que integran contenidos multimedia analizando y empleando las tecnologías y librerías específicas.
4. Selecciona y prueba motores de juegos analizando la arquitectura de juegos 2D y 3D.
5. Desarrolla juegos 2D y 3D sencillos utilizando motores de juegos.

Para esta unidad, se tendrá en cuenta el **resultado de aprendizaje 5,** cuyos **criterios de evaluación** son los siguientes:

a. Se ha establecido la lógica de un nuevo juego.
b. Se han creado objetos y definido los fondos.
c. Se han instalado y utilizado extensiones para el manejo de escenas.
d. Se han utilizado instrucciones gráficas para determinar las propiedades finales de la superficie de un objeto o imagen.
e. Se ha incorporado sonido a los diferentes eventos del juego.
f. Se han desarrollado e implantado juegos para dispositivos móviles.
g. Se han realizado pruebas de funcionamiento y optimización de los juegos desarrollados.
h. Se han documentado las fases de diseño y desarrollo de los juegos creados.

7.11 Metodología

En este apartado se describirá la metodología seguida en las actividades que se realizarán durante las unidades didácticas que componen la programación, además de algunos aspectos importantes a tener en cuenta como la disposición del aula o materiales necesarios.

Más adelante se especificará cuáles de esas actividades se realizarían en concreto en la unidad didáctica que se ha creado para este trabajo.

7.11.1 Organización del aula

Las clases se desarrollarán en el aula habitual asignada al curso de este módulo.

Como se ha comentado en anteriormente en otra sección (7.4.Instalaciones y materiales del centro), los puestos de trabajo cuentan con enchufes donde el alumno puede conectar su ordenador personal o, en caso de no tener, el ordenador que el centro le haya cedido.

Las mesas de este aula se disponen de forma individual en el espacio disponible, habiendo un puesto de trabajo para el profesor en la parte frontal junto a una pizarra

con rotuladores de colores y un proyector con pantalla para la visualización de diapositivas u otras herramientas visuales compartiendo la pantalla del ordenador (Figura 16).

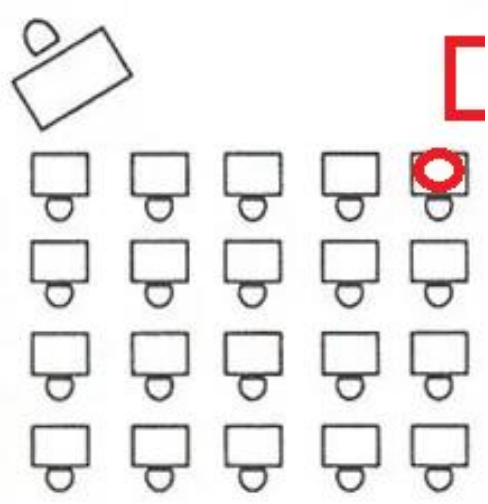

Figura 16: Disposición del aula. Fuente: Elaboración propia

Cabe destacar el puesto más cercano a la puerta irá destinado al alumno con discapacidades motoras (puesto señalado en rojo).

7.11.2 Tipos de agrupamiento

Según la metodología que se va a llevar a cabo, se utilizarán los siguientes agrupamientos para la realización de las actividades:

- **Trabajo individual:** Este tipo de agrupamiento será el que se utilice para actividades que tengan un mayor grado de personalización de la enseñanza, pudiendo así adaptarse la actividad al ritmo de cada uno de los alumnos. Esto puede resultar muy recomendable a la hora de afianzar ciertos conceptos. Además, el profesor tiene la oportunidad de hacer un seguimiento más cercano y detallado del proceso de aprendizaje de cada alumno. **Este agrupamiento será el habitual en la mayoría de actividades realizadas tras una explicación del contenido de esa sesión.**
- **Trabajo en parejas:** El uso de este tipo de agrupamiento permite al alumnado comunicarse entre ellos, compartir ideas y repartir el trabajo a realizar, siendo muy importante una participación activa de los dos alumnos. **Este tipo de agrupamiento se usará para algunas de las actividades prácticas en clase.**
- **Trabajo en grupo:** En el estudio de la informática, el trabajo en equipo resulta muy importante, ya que en la vida la boral de un informático , normalmente, se trabaja en equipos de trabajo. **Este tipo de**

agrupamiento se usará para proyectos y actividades más grandes, no superando los equipos un número de integrantes mayor a 5.

Dado que en el caso el grupo de alumnos cuenta con **20 personas**, tal y como se explica en el apartado dedicado al estudio del alumnado (7.3.Contexto del alumnado y entorno socio-cultural), los **equipos de trabajo serán de 4 o 5.**

7.11.3 Materiales y recursos didácticos

Los siguientes materiales y recursos serán necesarios para impartir la unidad didáctica creada:

1. Un ordenador por alumno que incluya Windows como sistema operativo u otro sistema operativo con software de virtualización.
2. Proyector y pantalla.
3. Pizarra y rotuladores de colores.
4. Programas de ofimática (LibreOffice)
5. Acceso a Internet.
6. Plataforma Moodle o alguna otra plataforma virtual que permita compartir recursos didácticos (actividades de clase, diapositivas, etc.).
7. Motor gráfico Unity.
8. Software de Visual Studio (gratuito) para la creación de scripts en C#.
9. Aplicaciones de edición multimedia para creación y modificación de imágenes, sonido y vídeo (Gimp, Audacity, etc.)

7.11.4 Metodología usada

La metodología que se usará para esta unidad didáctica tiene el fin de preparar al grupo de alumnos para su futura vida laboral, así como su vida personal y social. Por este motivo, la metodología que se seguirá tendrá principalmente un enfoque práctico que motive al alumno para la adquisición de los conocimientos necesarios.

Dicho esto, se seguirán una serie de directrices y estrategias que contribuirán a la adquisición del conocimiento y a la consecución de los objetivos de esta unidad:

- Se potenciará la autonomía, el pensamiento crítico, la reflexión y la creatividad del alumnado.
- Se pondrá al alumnado en situaciones diversas con el fin de desarrollar los diferentes procesos cognitivos tales como analizar problemas, identificar datos importantes, aplicar conocimientos adquiridos, etc.

- Se potenciará la independencia del alumno, debiendo ser capaz de plantearse sus propios objetivos y metas, organizarse y planificar el trabajo, buscar y seleccionar con una visión crítica la información necesaria, interpretar y contrastar resultados adquiridos, etc.
- Se intentarán contextualizar las actividades de modo que el alumno vea reflejada la aplicación de los conocimientos adquiridos en el mundo real y cercano a él.
- Se fomentará la investigación por parte del alumno y se utilizarán diversas fuentes de información y materiales donde el alumno pueda aprender a su estilo y ritmo.
- Se promoverá el trabajo de forma colaborativa, buscando siempre una heterogeneidad en los grupos de trabajo. De esta forma se busca un reflejo de la actual sociedad donde el alumno tendrá que hacer su vida laboral en un futuro, preparándolo así para ello. Con esto también se pretende fomentar el respeto, la empatía y la solidaridad entre los diferentes alumnos.

7.11.5 Actividades de Enseñanza-Aprendizaje

Se mostrarán a continuación los diferentes tipos de actividades de enseñanza-aprendizaje que se utilizarán de acuerdo a la metodología anteriormente explicada y que se usarán para el desarrollo de las actividades programadas:

- **Actividad de introducción:** Estas actividades tienen el fin de introducir al alumnado la sesión en cuestión. En el caso de una nueva UD, en este tipo de actividad se realizará una pequeña evaluación inicial y se introducirán los conceptos que se darán en esta y se explicarán algunos aspectos relativos a los objetivos y evaluación de dicha UD. En el caso de una UD ya empezada, se recordarán conceptos previos de dicha UD y se relacionarán con los que se van a explicar en la sesión actual.
- **Actividad de desarrollo:** En este tipo de actividades se procurará explicar los contenidos propuestos para esa sesión con el fin de conseguir los objetivos de la UD. En este caso, este tipo de actividad será una clase magistral donde el profesor utilizará los recursos disponibles para explicar los contenidos de la UD.
- **Actividad de consolidación:** Este tipo de actividades serán necesarias a la hora de aplicar los contenidos explicados, resolviéndose problemas

prácticos de forma individual. Estas actividades serán clave para el aprendizaje de los conocimientos explicados.

- **Actividad de resumen/diagnóstico:** Estas actividades tendrán como finalidad el seguimiento del aprendizaje por parte del alumnado, haciendo algún pequeño cuestionario o actividad que resuma el contenido visto en la sesión y algunos conceptos dados en sesiones anteriores de dicha UD. Estas se encontrarán al final de la mayoría de las sesiones.
- **Actividad de proyecto:** En esta actividad el alumnado desarrollará un proyecto algo más amplio donde se tendrá que aplicar todo lo visto en la UD. Este tipo de actividad se hará normalmente en equipos de trabajo en los que se procurará que haya un **equilibro entre número de hombres y mujeres (Igualdad de género)**. **Para esta unidad, el proyecto tendrá una temática basada en las energías renovables, de forma que se relacione con dicho tema transversal.**
- **Actividad de exposición:** Este tipo de actividad tiene como finalidad la exposición por parte de los alumnos del contenido creado en la actividad de proyecto. Esta actividad será llevada a cabo por grupos, los mismos creados en la actividad de proyecto.
- **Actividad de evaluación:** Con este tipo de actividad se evaluarán de forma individual los conceptos teóricos explicados en las actividades de desarrollo que el alumno ha aprendido.
- **Recuperación:** Esta actividad se llevará a cabo en el caso de que alguno de los alumnos no haya superado los objetivos que plantea la unidad didáctica, teniendo que hacer una prueba o trabajo individual.

7.11.6 Temporización de actividades

Como se ha mencionado anteriormente, esta unidad didáctica se desarrollará a lo largo de 8 sesiones de 2 horas cada una (16 horas en total).

Todas las actividades propuestas se desarrollarán en el aula, teniendo que completarse excepcionalmente en casa en el caso de que el alumno no haya podido completar a tiempo alguna actividad de consolidación o el proyecto.

Se puede ver la temporización de las actividades propuestas para esta UD en la tabla de temporización de actividades (Tabla 2):

Temporización de actividades			
Recursos	Ordenador, Motor gráfico Unity	**Espacio**	Aula
Sesión 1			
Tipo Act.	**Duración**	**Descripción**	
Introducción	20 min	Pequeña evaluación inicial e introducción a la unidad y a los conceptos dados en ella. Se explicarán aquí los objetivos y evaluación de los mismos.	
Desarrollo	50 min	Explicación de las fases del desarrollo de videojuegos e introducción a Unity: • Fases de desarrollo de un juego • Instalación de Unity. • Creación de un nuevo proyecto. • Configuración del proyecto. • Toma de contacto con la interfaz del editor de Unity. • Creación de GameObjects básicos en la escena.	
Consolidación	40 min	Crear un nuevo proyecto. Dentro de este, crear una escena en Unity con GameObjects diferentes	
Resumen / diagnóstico	10 min	Test para resumir lo dado en la sesión y resolución de dudas.	
Sesión 2			
Tipo Act.	**Duración**	**Descripción**	
Introducción	10 min	Repaso de conceptos anteriores y breve introducción a lo que se hará en esta sesión	
Desarrollo	40 min	Explicación de nuevos conceptos: • Creación de reglas de juego. Condiciones de victoria y derrota. • Añadir un personaje (ya creado en Unity). • Propiedad transform (Posición, rotación y escalado). • Detección de colisiones.	
Consolidación	60 min	Crear una fase de un nivel añadiendo los GameObjects necesarios. Se determinará una zona o elemento del nivel como “meta” y otra como “fracaso”. Se detectarán colisiones con estos y, en caso de que el jugador colisione con la “meta”, habrá ganado. Si colisiona con “fracaso” habrá perdido.	
Resumen / diagnóstico	10 min	Test para resumir lo dado en la sesión y resolución de dudas.	
Sesión 3			
Tipo Act.	**Duración**	**Descripción**	
Introducción	10 min	Repaso de conceptos anteriores y breve introducción a lo que se hará en esta sesión	
Desarrollo	40 min	Explicación de nuevos conceptos: • Mecánicas de un juego. • Dar movimiento mediante scripts. • Control de cámara. • Crear una interfaz.	
Consolidación	60 min	Creación de un nivel con enemigos móviles y controlar el giro de la cámara.	
Resumen / diagnóstico	10 min	Test para resumir lo dado en la sesión y resolución de dudas.	
Sesión 4			
Tipo Act.	**Duración**	**Descripción**	
Introducción	10 min	Repaso de conceptos anteriores y breve introducción a lo que se hará en esta sesión	
Desarrollo	50 min	Explicación de nuevos conceptos (50 min): • Modos de juego.	

		• Control de capas. • Raycasting (Detección de objetos con rayos). • Efectos de sonido.
Consolidación	50 min	Crear un nivel donde se destruyen enemigos con un disparo del jugador. Añadir efectos de sonido al disparar y morir un enemigo
Resumen / diagnóstico	10 min	Test para resumir lo dado en la sesión y resolución de dudas.
Sesión 5		
Tipo Act.	**Duración**	**Descripción**
Introducción	10 min	Repaso de conceptos anteriores y breve introducción a lo que se hará en esta sesión
Desarrollo	50 min	Explicación de nuevos conceptos: • Diagrama de flujo del juego. • Guardar información entre escenas diferentes. • Iluminación. Tipos de luces. • Creación de materiales. • Aplicación de texturas.
Consolidación	40 min	Modificar la actividad de consolidación de la anterior sesión. con la aplicación de materiales, texturas y luces. Crear otro nivel similar y guardar información como una “puntuación” entre las escenas.
Proyecto	10 min	Explicación de la actividad de proyecto en grupos (creación de un videojuego propio con temática de las energías renovables).
Resumen / diagnóstico	10 min	Test para resumir lo dado en la sesión y resolución de dudas.
Sesión 6		
Tipo Act.	**Duración**	**Descripción**
Introducción	10 min	Repaso de conceptos anteriores y breve introducción a lo que se hará en esta sesión
Desarrollo	25 min	Explicación de nuevos conceptos: • Descarga de nuevos assets desde la tienda. • Creación de animaciones.
Proyecto	75 min	Trabajo en grupo para la creación del proyecto (creación de un videojuego propio).
Resumen/diagnóstico	15 min	Resolución de dudas.
Sesión 7		
Tipo Act.	**Duración**	**Descripción**
Introducción	10 min	Repaso de conceptos anteriores y breve introducción a lo que se hará en esta sesión
Proyecto	95 min	Trabajo en grupo para la creación del proyecto (creación de un videojuego propio).
Resumen / diagnóstico	15 min	Resolución de dudas.
Sesión 8		
Tipo Act.	**Duración**	**Descripción**
Exposición	90 min	Exposición del proyecto creado en grupos. 15 min para cada grupo (4 grupos) y tiempo extra para imprevistos.
Evaluación	30 min	Test de evaluación de conceptos teóricos.

Tabla 2: Temporización de actividades

7.11.7 Bibliografía de aula

- Manual de Unity hecho por el profesor (10.2 Manual de Unity para clase).
- Manual de Unity online (Unity Technologies, s. f.).
- Libro "Desarrollo de videojuegos: un enfoque práctico" (Vallejo & Cleto, 2015).
- Libro "Game Engine Architecture" (Gregory, 2009).

7.12 Evaluación

Se debe llevar a cabo la evaluación del alumnado para conocer el grado de conocimiento que han alcanzado los alumnos en relación a los objetivos dados. De esta manera, también será importante determinar si la forma de enseñanza ha sido la adecuada para alcanzar estos objetivos. Dicha evaluación, de acuerdo con el Real Decreto 450/2010, de 16 de abril, será continua, formativa y sumativa.

7.12.1 Instrumentos de evaluación

Para la evaluación del alumnado, en esta unidad se usarán los instrumentos de evaluación que se han considerado oportunos para ello (Tabla 33).

Instrumento de Evaluación	Descripción	Peso	Temporización
Actitud y participación	Observación directa por parte del profesor, la actitud del alumno en clase, su participación en las sesiones, el cumplimiento de las normas del centro, etc.	10%	Durante la UD
Actividades	Actividades de consolidación que se proponen al alumnado tras la explicación de contenidos	25%	Durante la UD
Proyecto	Proyecto elaborado por grupos de trabajo en clase y en casa	30%	Durante la UD
Exposición	Exposición en grupos del trabajo realizado en el proyecto y sus resultados	15%	Último día de la UD
Examen	Actividad de evaluación que consiste en la resolución de una prueba teórica al final de la UD	20%	Último día de la UD

Tabla 3: Instrumentos de evaluación

7.12.2 Ponderación de los criterios de evaluación

Para una correcta evaluación del alumnado es necesario establecer una ponderación de los diferentes criterios de evaluación que serán tenidos en cuenta para esta unidad didáctica (Tabla 44).

Para mayor información, se han clasificado dichos criterios según su importancia y se han relacionado estos con las CPPS oportunas.

RA	CE	CPPS	Consideración	Peso
5	a	d, h, i	Imprescindible	15%
	b	d, g, i, j	Imprescindible	15%
	c	d, i, j	Deseable	10%
	d	d, g, h, i, j	Imprescindible	15%
	e	d, g, h, i, j	Imprescindible	10%
	f	d, g, h, i, j, m, t	Imprescindible	15%
	g	d, h, i, j, t	Imprescindible	10%
	h	h, i, j, t	Deseable	10%

Tabla 4: Ponderación de los criterios de evaluación

7.12.3 Otras consideraciones

Se llevará a cabo una evaluación inicial al inicio de cada unidad que consistirá en la puesta en común de conocimientos previos de los alumnos en la primera sesión de esta.

La evaluación continua se tendrá en cuenta a través de la observación directa hacia el alumnado a la hora de realizar las actividades de consolidación tras la actividad de desarrollo del contenido que corresponda a dicha sesión. Además de esto, se comprobará el trabajo que realiza el alumno en el proyecto que tendrá que llevar a cabo.

La evaluación final se hará a través de una prueba teórica tipo test con conceptos que tendrán relación con lo trabajado en la UD.

Para la superación de dicha unidad, será necesario el aprobado con un mínimo de un 5 haciendo la media ponderada de todos los instrumentos de evaluación, teniendo en cuenta que el alumno deberá de tener una puntuación mínima de 5 en la parte del proyecto y actividades, así como un mínimo de 4 en la prueba teórica.

7.12.1 Recuperación

En caso de haber alumnos con una calificación insuficiente para la superación de la unidad, el alumno realizará un trabajo práctico y su correspondiente defensa que resuma todo lo dado en la UD, pudiendo entonces optar a una nota máxima de un 7 en la nota final de esta UD.

7.12.2 Evaluación de la enseñanza

El proceso de evaluación no se debe aplicar solo al alumnado, sino que es necesaria también la evaluación de la labor docente para poder mejorar la docencia en cursos posteriores. De esta forma se pueden detectar fallos a la hora de transmitir la información y mejoras que el docente puede aplicar para que la adquisición de conocimiento por parte del alumnado sea adecuada.

En el caso de la programación donde se enmarca esta UD, la evaluación del proceso de enseñanza se realizará mediante un cuestionario que el profesor repartirá en las últimas sesiones. Véase la Tabla 6: Cuestionario para la evaluación de la enseñanza que se encuentra en el apartado 12. Anexos.

7.13 Tabla resumen de la UD

En la tabla se muestra toda la información resumida de la unidad didáctica.

<table>
<tr><td colspan="3">UD 5: Desarrollo de videojuegos para plataformas móviles</td><td colspan="2">Desde</td><td>25/02</td><td>Hasta</td><td>14/03</td></tr>
<tr><td colspan="3"></td><td colspan="2">% Curso</td><td>20%</td><td>% Trim</td><td>60%</td></tr>
<tr><td>RA</td><td colspan="7">5. Desarrolla juegos 2D y 3D sencillos utilizando motores de juegos.</td></tr>
<tr><td>Horas</td><td>2h x 8 ses.</td><td>OG</td><td>h, i, j, s</td><td>OP</td><td>3</td><td>LA</td><td>4,5,6</td><td>CPPS</td><td>d, g, h, i, j, m, t.</td></tr>
<tr><td colspan="5">Objetivos específicos</td><td colspan="5">Contenidos</td></tr>
<tr><td colspan="5">1. Conocer el motor de videojuegos Unity
2. Integrar el uso de Unity en entornos de desarrollo
3. Utilizar conceptos avanzados de programación 2D y 3D
4. Conocer las fases de desarrollo de un videojuego
5. Conocer y manejar propiedades de los objetos, luz, texturas, reflejos y sombras.
6. Conocer y aplicar las funciones del motor gráfico Unity.
7. Aplicar las funciones del grafo de escena.
8. Analizar, documentar y optimizar el código creado.</td><td colspan="5">A. Entornos de desarrollo para juegos.
B. Integración del motor de juegos en entornos de desarrollo.
C. Conceptos avanzados de programación 3D.
D. Fases de desarrollo.
E. Propiedades de los objetos, luz, texturas, reflejos, sombras.
F. Aplicación de las funciones del motor gráfico. Renderización.
G. Aplicación de las funciones del grafo de escena. Tipos de nodos y su utilización.
H. Análisis de ejecución. Optimización del código.</td></tr>
<tr><td colspan="10">Elementos transversales y educación en valores</td></tr>
<tr><td colspan="10">Igualdad de género. Energías renovables.</td></tr>
<tr><td colspan="10">Evaluación</td></tr>
<tr><td>Instr. Eval.</td><td>%</td><td>CE</td><td>%</td><td colspan="4">Contenidos</td><td colspan="2">Obj. Específicos</td></tr>
<tr><td rowspan="2">Actitud y participación</td><td rowspan="2">10%</td><td>a</td><td>15%</td><td colspan="4">A, B, D</td><td colspan="2">1, 2, 4</td></tr>
<tr><td rowspan="2">b</td><td rowspan="2">15%</td><td colspan="4" rowspan="2">A, B, C, F, G</td><td colspan="2" rowspan="2">1, 2, 3, 6, 7</td></tr>
<tr><td rowspan="2">Actividades</td><td rowspan="2">25%</td></tr>
<tr><td>c</td><td>10%</td><td colspan="4">A, C, F, G</td><td colspan="2">1, 3, 6, 7</td></tr>
<tr><td rowspan="2">Proyecto</td><td rowspan="2">30%</td><td>d</td><td>15%</td><td colspan="4">A, B, C, E, F</td><td colspan="2">1, 2, 3, 5, 6</td></tr>
<tr><td>e</td><td>10%</td><td colspan="4">A, F</td><td colspan="2">1, 6</td></tr>
<tr><td>Exposición</td><td>15%</td><td>f</td><td>15%</td><td colspan="4">A, B, C, D, E ,F, G, H</td><td colspan="2">1, 2, 3, 4, 5, 6, 7, 8</td></tr>
<tr><td rowspan="2">Examen</td><td rowspan="2">20%</td><td>g</td><td>10%</td><td colspan="4">H</td><td colspan="2">8</td></tr>
<tr><td>h</td><td>10%</td><td colspan="4">B, D, H</td><td colspan="2">2, 4, 8</td></tr>
</table>

Tabla 5: Tabla resumen de la UD

8. Atención a la diversidad

8.1 Actividades de refuerzo/ampliación

Se contemplarán actividades de refuerzo para los alumnos que presenten dificultades a la hora de comprender los contenidos, como la realización de pequeños ejercicios guiados paso a paso que sirvan para la obtención total de los conocimientos necesarios.

En el caso de alumnos aventajados, se harán responsables de explicar y ayudar a otros compañeros a los que les cueste más. Además, se les propondrá la realización de ejercicios más complejos.

8.2 Necesidades educativas especiales

Dado la dificultad de acceso de uno de los alumnos que cuenta con silla de ruedas, se dejará siempre a dicho alumno cerca de la puerta para que tenga la menor dificultad de acceso posible a su puesto en la clase.

En cualquier caso, se trabajará en contacto continuo con el departamento de orientación del centro.

9. Conclusiones

Tras el desarrollo de esta unidad didáctica, el alumnado será capaz de desarrollar por sí mismos juegos en 2D y 3D para plataformas móviles, despertando la curiosidad para la realización de proyectos aún más grandes. Además, con el trabajo colaborativo se lleva a los alumnos a una situación más realista dado que el trabajo de desarrollador se realiza, de manera general, en grupos de trabajo muy heterogéneos.

Como profesor que imparte esta unidad didáctica, el tema elegido resulta realmente interesante y ameno, pudiendo observar la creatividad del alumnado con la gran libertad que el desarrollo de un videojuego puede dar.

10. Anexos

10.1 Cuestionario para la evaluación de la enseñanza

Nombre Profesor:		Fecha:	
Curso:		Materia:	

Marca del 1 al 5 estas afirmaciones según tu grado de acuerdo (1=Totalmente en desacuerdo, 2=En desacuerdo, 3=Neutral, 4=De acuerdo, 5=Totalmente de acuerdo, NS/NC=No sabe/No contesta):

	1	2	3	4	5	NS/NC
1. Tengo un gran interés por lo que se da en esta asignatura						
2. Le dedico el tiempo suficiente a estudiar esta asignatura						
3. Asisto regularmente a clase						
4. El profesor explica las cosas con claridad						
5. El profesor se interesa por el grado de interés de los alumnos						
6. El profesor resuelve las dudas que se le plantean						
7. El profesor transmite seguridad a la hora de explicar						
8. El profesor promueve un buen clima de trabajo						
9. Siento apoyo por parte del profesor						
10. Pienso que el método de evaluación es adecuado						
11. Creo que esta asignatura me puede ayudar en mi futuro						
12. El profesor cumple con los horarios						
13. El profesor deja claro qué va a evaluar y cómo lo va a hacer						
14. Trabajo en casa lo suficiente para aprobar esta asignatura						
15. Hay una buena organización de la asignatura						
16. El profesor tiene un trato adecuado con los alumnos						
17. Esta encuesta la estoy haciendo a voleo						
18. Cursar esta asignatura es importante para mi futuro						
19. Pienso que es un buen profesor						
20. Estoy satisfecho con la labor docente del profesor						

Observaciones:

Tabla 6: Cuestionario para la evaluación de la enseñanza

10.2 Manual de Unity para clase

Este manual breve se les dará a los alumnos para comenzar a trabajar con el motor gráfico de Unity. Aun así, se les dará acceso también al manual propio de Unity (Unity Technologies, s. f.).

10.2.1 Primeros pasos

Para comenzar a programar en Unity, será necesario descargar la versión deseada para nuestro desarrollo. Es recomendable usar las versiones LTS o Long Term Support, ya que son las versiones más estables que reciben soporte durante más tiempo, aunque se pueden llegar a usar otras versiones para probar nueva funcionalidad en la aplicación. Algo muy importante a la hora de la instalación de nuestro editor es la elección de los paquetes a instalar, ya que, si queremos desarrollar juegos para móviles, por ejemplo, debemos descargar el paquete de Android SDK.

Una vez descargado e instalado, podremos empezar nuestro proyecto.

10.2.2 Comenzando el proyecto

A la hora de comenzar con nuestro proyecto, podemos empezar un proyecto en blanco, aunque también puede ser apropiado empezar con alguna de las plantillas que nos ofrece Unity si nuestro proyecto en mente se parece a alguna de estas (Figura 17).

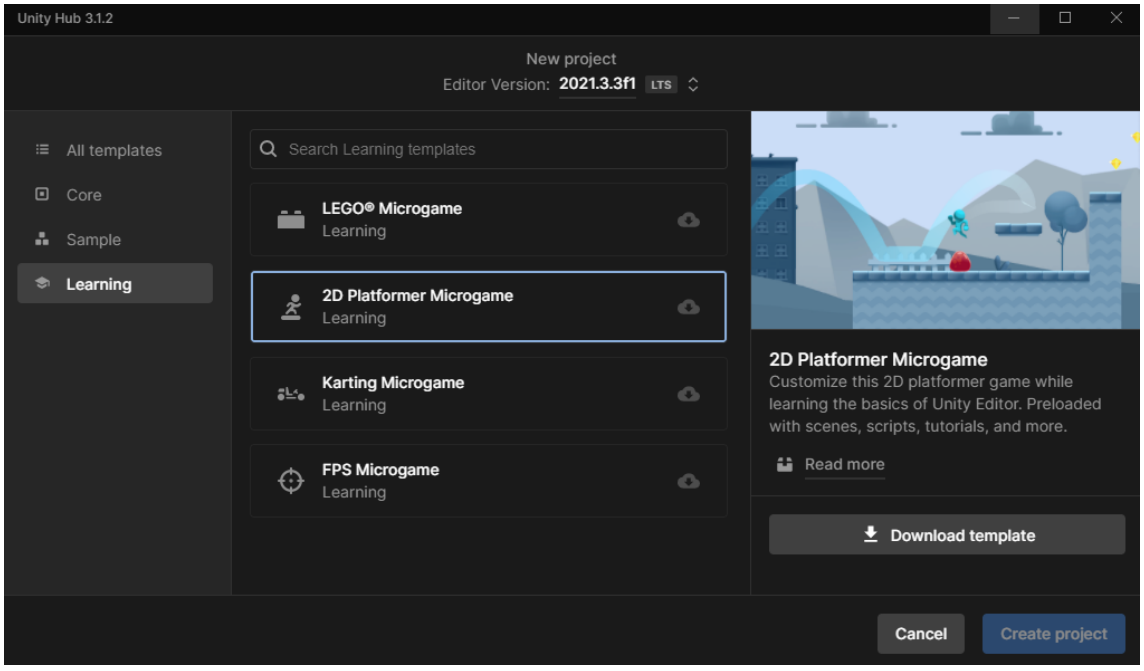

Figura 17: Crear proyecto en Unity. Fuente: Elaboración propia

Una vez creado el proyecto, se abrirá el editor de Unity donde podremos ver la interfaz de Unity junto con la escena y los diferentes elementos disponibles (Figura 18).

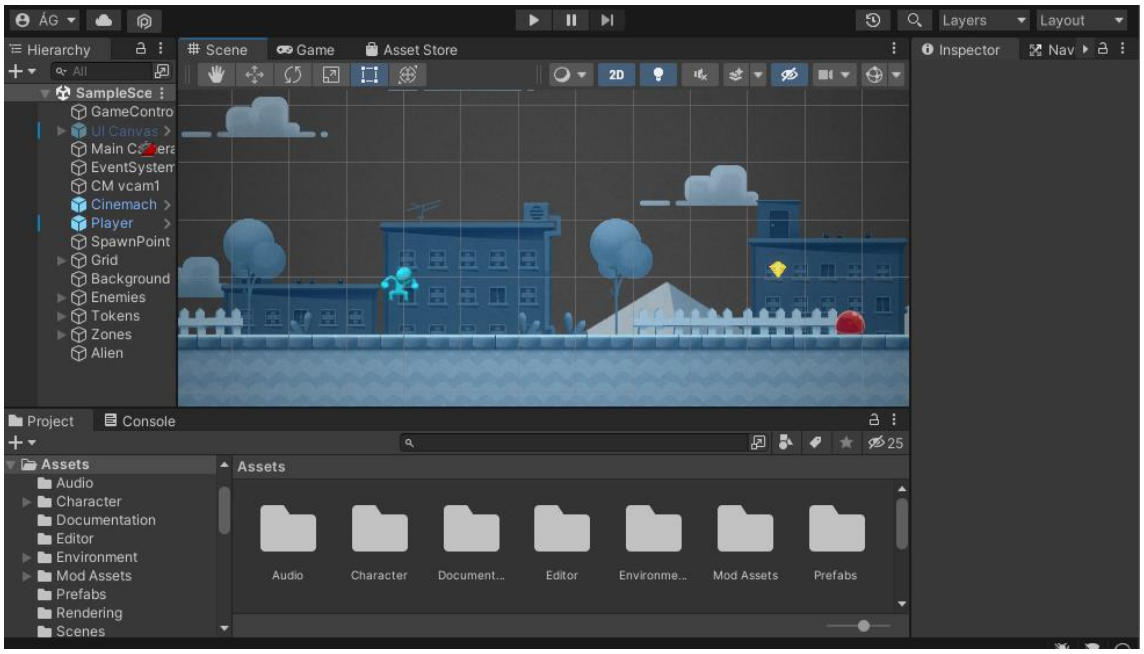

Figura 18: Interfaz de Unity. Fuente: Elaboración propia

10.2.3 Elementos importantes de Unity

Una vez que hemos creado el proyecto, se debe trabajar ya en el videojuego, añadiendo elementos al mismo y programando todo lo necesario para crear todo aquello que se quiera.

A continuación, se expondrán de forma breve aquellos elementos que son fundamentales en la creación de un videojuego en Unity.

10.2.3.1 La escena

La escena en Unity no es solo el espacio de juego, sino que es aquella que guardará todos los elementos necesarios (GameObjects) para la creación de un nivel o escenario específico, incluidos todos aquellos que no se muestran en pantalla.

Podemos observar en la Figura 19, como la escena “SampleScene” contiene todos los elementos de ese nivel.

Figura 19: Jerarquía en la escena. Fuente: Elaboración propia

10.2.3.2 GameObjects

Los GameObjects son los objetos que usa Unity para guardar cada una de las entidades que conforman la escena.

Estos GameObjects tienen propiedades que definen a ese objeto como lo que es y que lo diferencia de otros. En definitiva, los GameObjects son contenedores que guardan las "piezas" necesarias para la construcción de un objeto como una luz, un árbol, un sonido o al mismo personaje principal.

10.2.3.3 Componentes

Decimos que Unity utiliza una programación orientada o basada en componentes. Estos componentes son las "piezas" que mencionábamos anteriormente y son las que le dan unas características concretas a nuestros GameObjects (Figura 20).

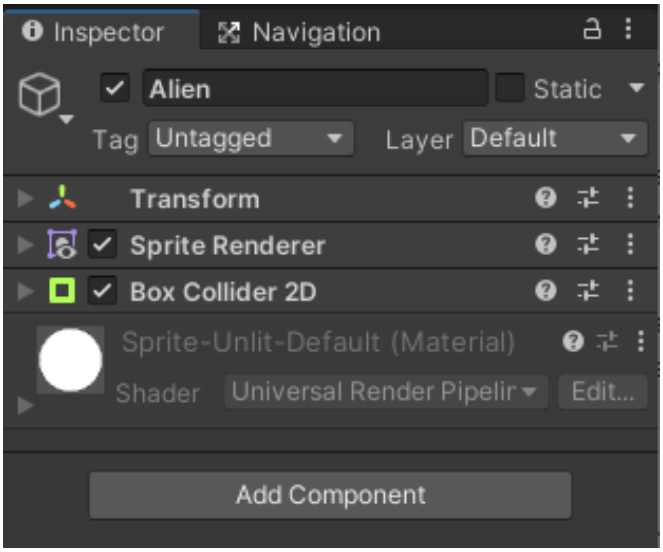

Figura 20: Componentes de un GameObject. Fuente: Elaboración propia

Algunos de los componentes más importantes que se pueden agregar a un GameObject son los siguientes:

- **Transform**: Se usa para dar una posición, una rotación y un factor de escala al GameObject en cuestión (Figura 21). Todos los GameObjects se crean con este componente ya incluido por defecto.

Figura 21: Componente Transform. Fuente: Elaboración propia

- **Rigidbody**: Este componente añade una masa al GameObject, de manera que, una vez agregado, puede ser controlado por el motor de física de Unity, haciendo que sienta fuerzas, como la de la gravedad, así como colisiones que puedan afectarle (Figura 22).

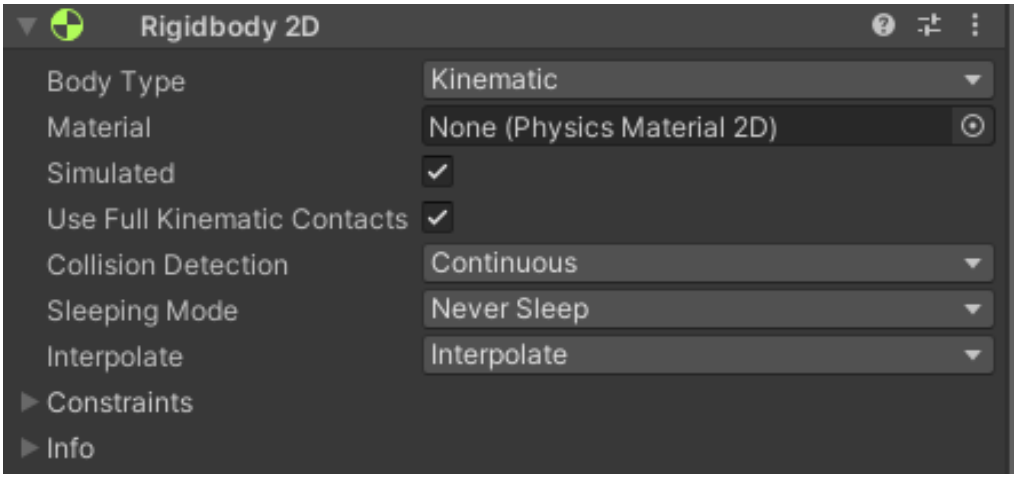

Figura 22: Componente Rigidbody. Fuente: Elaboración propia

- **Audio Source**: Usado para añadir a ese GameObject un clip de audio (Figura 23). Este clip se puede manejar según las propiedades del componente, así como por scripts.

Figura 23: Componente Audio Source. Fuente: Elaboración propia

- **Cámera**: Añade al GameObject una cámara (Figura 24). Las cámaras son necesarias para visualizar el mundo del videojuego cuando se ejecuta este. Por defecto, Unity crea las escenas con un GameObject que ya incluye una cámara para poder ver la escena si se prueba esta dándole al play.

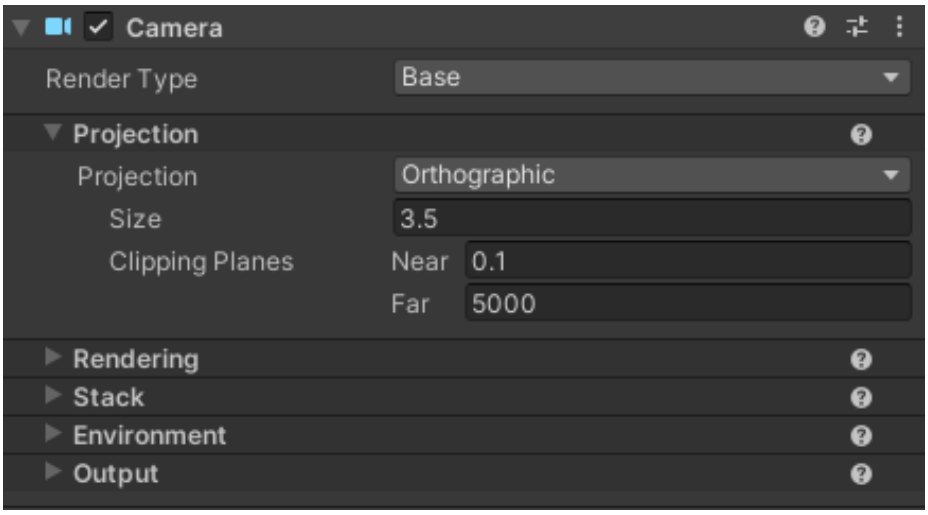

Figura 24: Componente Cámera. Fuente: Elaboración propia

- **Collider**: Este “colisionador” establece un área alrededor del GameObject en cuestión que se usa para la detección de colisiones (Figura 25). Normalmente, estos son una aproximación al objeto que los rodea, siendo mucho más sencillo detectar la colisión en tiempo real.

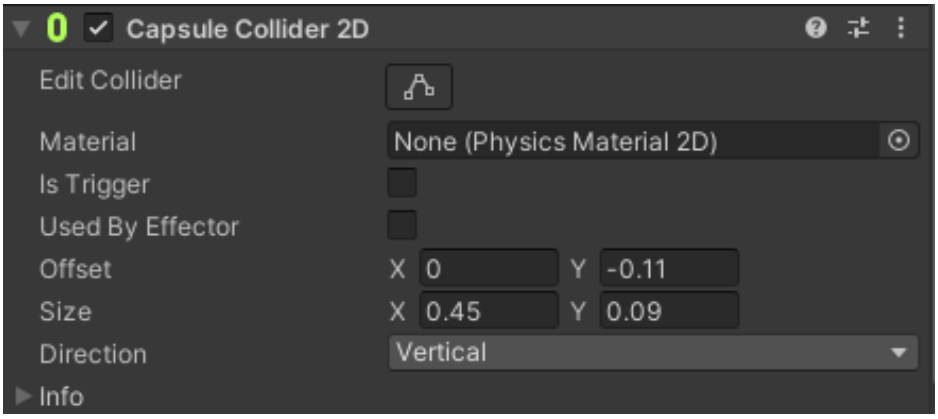

Figura 25: Componente Collider. Fuente: Elaboración propia

- **Light**: Este componente añade una luz al GameObject (Figura 26). Las luces son componentes esenciales en la escena ya que conforman la iluminación de la escena.

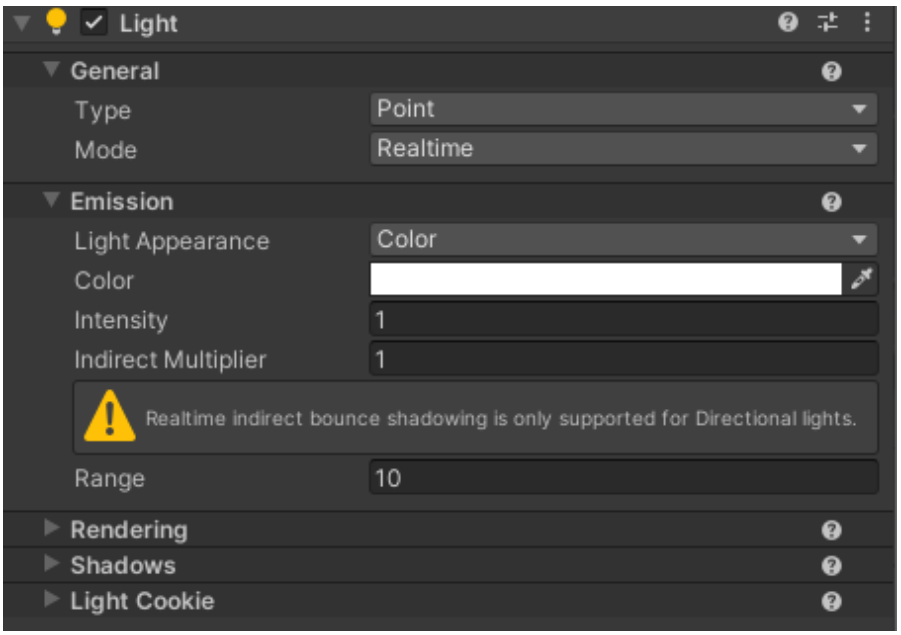

Figura 26: Componente Light. Fuente: Elaboración propia

10.2.3.4 Prefabs

Un Prefab (un objeto prefabricado) es aquel objeto que se utiliza para construir GameObjects a partir del mismo. De esta forma, se puede decir que un Prefab es una plantilla que se utilizará para instanciar objetos repetidos en la escena, sin necesidad de crearlos uno por uno.

Esto permite dos cosas:

- Instanciar GameObjects con unas propiedades concretas de forma más sencilla a través de scripting.
- Modificar al mismo tiempo todos los GameObjects creados a partir de un Prefab modificando solo las propiedades del Prefab.

10.2.4 Materiales

Uno de los elementos más importantes a la hora de crear gráficos por ordenador es el estudio de las propiedades de los materiales, ya que es importante dar a los elementos 2D y 3D la apariencia del material del que se supone que están hechos (en caso de querer unos gráficos realistas).

En Unity, como en otros motores gráficos, se pueden crear materiales propios para luego asignarlos a los elementos de la escena (Figura 27).

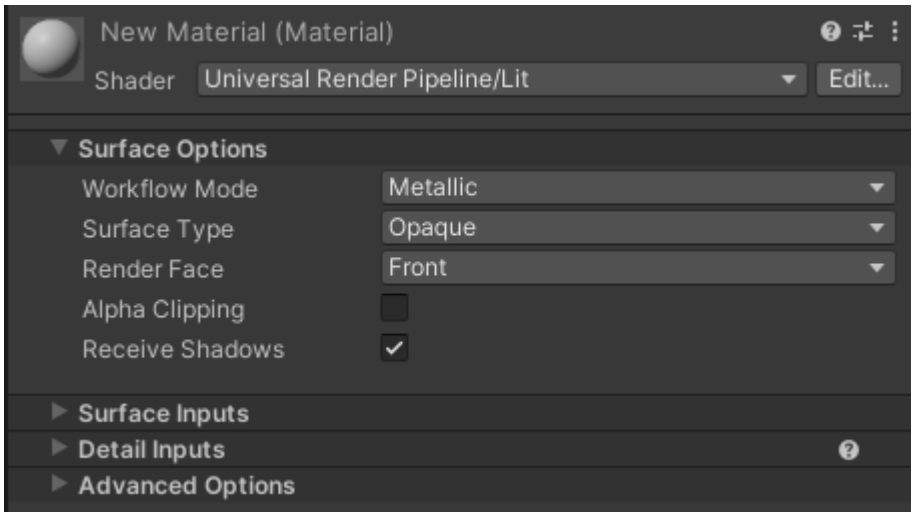

Figura 27: Nuevo material en Unity. Fuente: Elaboración propia

Unity permite configurar algunas opciones para crear materiales nuevos como:

- La iluminación de la superficie
- El tipo de superficie (opaca o transparente)
- Si recibe o no sombras
- El color de la base del material

- El color de la luz que se emite al rebotar la luz de la escena en el material
- Otros opciones avanzadas y de detalle

Una vez creado el nuevo material, aplicarlo es tan sencillo como arrastrarlo hasta el GameObject que se quiera en la jerarquía de elementos de la escena.

10.2.5 Scripting

Una parte fundamental en el desarrollo de un videojuego es la programación del mismo. Los elementos de la escena deben de tener asociados ciertos scripts que permitan el control de la acción en la escena.

Unity usa el lenguaje de programación C# y la herramienta Visual Studio de Microsoft para la creación del código necesario para programar nuestro videojuego. También puede usarse el lenguaje propio de Unity, el lenguaje UnityScript, modelado tras JavaScript. Una tercera opción es la de programar en lenguajes .NET siempre que se compilen con un DLL compatible.

```
Mensaje de Unity | 0 referencias
void Awake()
{
    health = GetComponent<Health>();
    audioSource = GetComponent<AudioSource>();
    collider2d = GetComponent<Collider2D>();
    spriteRenderer = GetComponent<SpriteRenderer>();
    animator = GetComponent<Animator>();
}

Mensaje de Unity | 2 referencias
protected override void Update()
{
    if (controlEnabled)
    {
        move.x = Input.GetAxis("Horizontal");
        if (jumpState == JumpState.Grounded && Input.GetButtonDown("Jump"))
            jumpState = JumpState.PrepareToJump;
        else if (Input.GetButtonUp("Jump"))
        {
            stopJump = true;
            Schedule<PlayerStopJump>().player = this;
        }
    }
    else
    {
        move.x = 0;
    }
    UpdateJumpState();
    base.Update();
}
```

Figura 28: Ejemplo de script en Unity. Fuente: Elaboración propia

Como se puede ver en la Figura 28, se suelen utilizar los siguientes métodos para controlar el script:

- **Awake**: Este método es el que se ejecuta una vez que aparece en la escena el GameObject al que está asociado el script. Aquí se indicarán las acciones y configuración inicial del objeto.
- **Update**: Este método se ejecuta una vez por cada frame. Suele incluir acciones como movimiento o respuestas al input del usuario. En resumen, controla cualquier factor que se necesite manejar durante el gameplay.

10.2.6 Animaciones

En Unity se puede controlar de forma fácil las animaciones de los personajes mediante Animator y el sistema de animación que incluye, Mecanim (Figura 29).

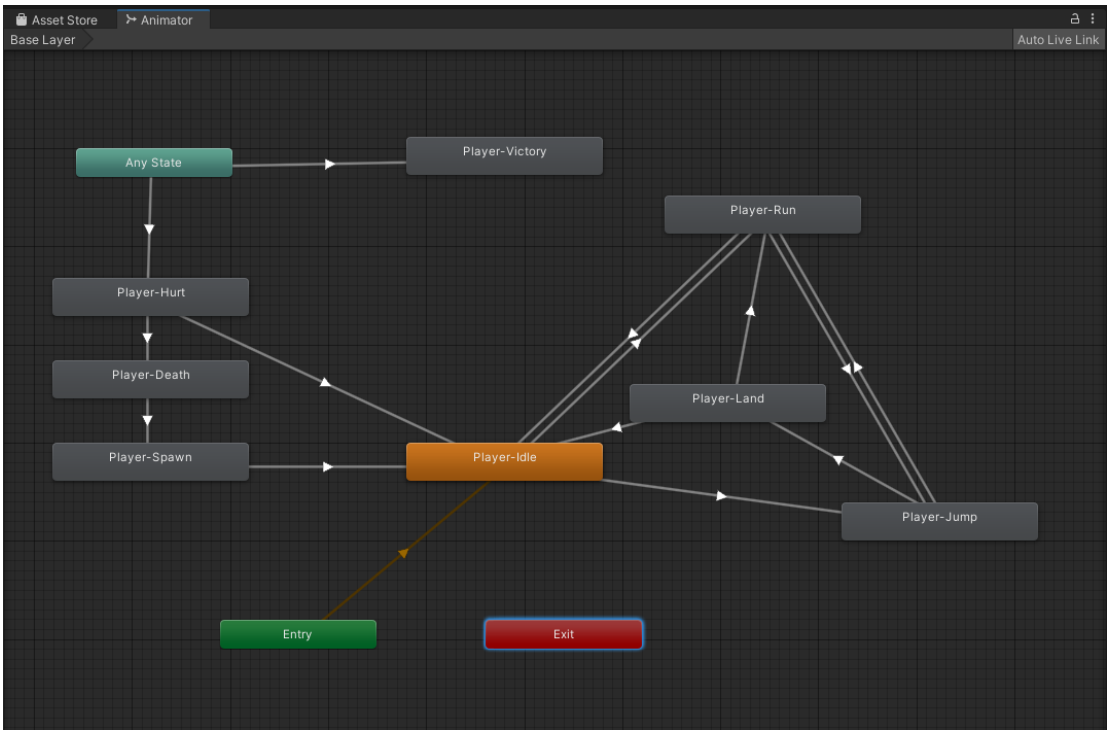

Figura 29: Herramienta Animator de Unity. Fuente: Elaboración propia

Este sistema de animación proporciona:

- Un flujo de trabajo y configuraciones de animaciones para todos los elementos de Unity incluyendo objetos, personajes, y propiedades de forma fácil, rápida y visual.
- Clips de animaciones importados
- Animación humanoide retargeting (lo que permite aplicar las animaciones de un modelo a otro diferente)
- Un flujo de trabajo simplificado para alinear animaciones.
- Una previsualización animaciones, transiciones e interacciones entre ellos, lo que permite al animador trabajar de forma independiente a los programadores sin necesidad de que el juego esté en funcionamiento.
- Animar diferentes partes del cuerpo con diferente lógica.
- Características de capas (layering) y de masking.

10.2.7 Configuración del proyecto

A la hora de crear nuestro proyecto, será necesario tener en cuenta algunas configuraciones importantes.

Para abrir la configuración del proyecto hay que abrir el submenú Edit > Project Settings. Tras esto se abrirá una pantalla desde la que se pueden configurar muchos de los aspectos del proyecto (Figura 30).

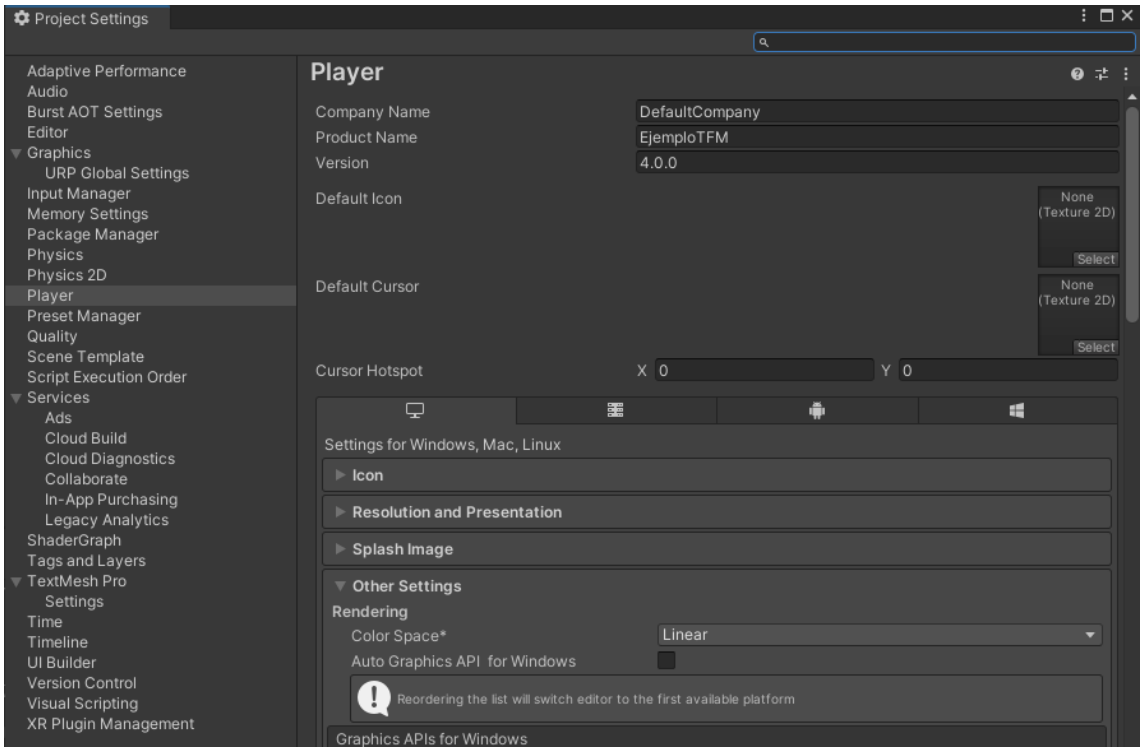

Figura 30: Configuración del proyecto. Fuente: Elaboración propia

En esta ventana, algunas de las configuraciones que podemos cambiar son:

- **Audio:** En esta pestaña se puede modificar el volumen del sonido, efectos, modos (estéreo, mono, etc.) entre otras cosas.
- **Gráficos:** En esta pestaña se podrán configurar aspectos relacionados con los shaders, luces, niebla, etc.
- **Físicas:** Unity permite también configurar el sistema de físicas modificando opciones como la gravedad, el rebote y la fricción entre otras opciones. Cabe destacar que podemos modificar esto de forma independiente para físicas en 2D.
- **Ejecutable (Player):** Esta pestaña es una de las más importantes a la hora de configurar el proyecto. En esta pestaña se podrán configurar aspectos como el nombre de la compañía desarrolladora, el nombre del juego o ejecutable, la versión del mismo y el icono del ejecutable. También se

pueden modificar opciones como el modo de pantalla (completa o ventana), resoluciones de la pantalla, si se permite la ejecución en segundo plano, etc.

Aunque hay más configuraciones disponibles, las mencionadas pueden ser algunas de las más importantes a la hora de configurar nuestro proyecto.

10.2.8 Compilar el proyecto y exportar

Para configurar la compilación de nuestro proyecto hay que dirigirse a File > Build Settings para abrir la pantalla de configuración del proyecto (Figura 31).

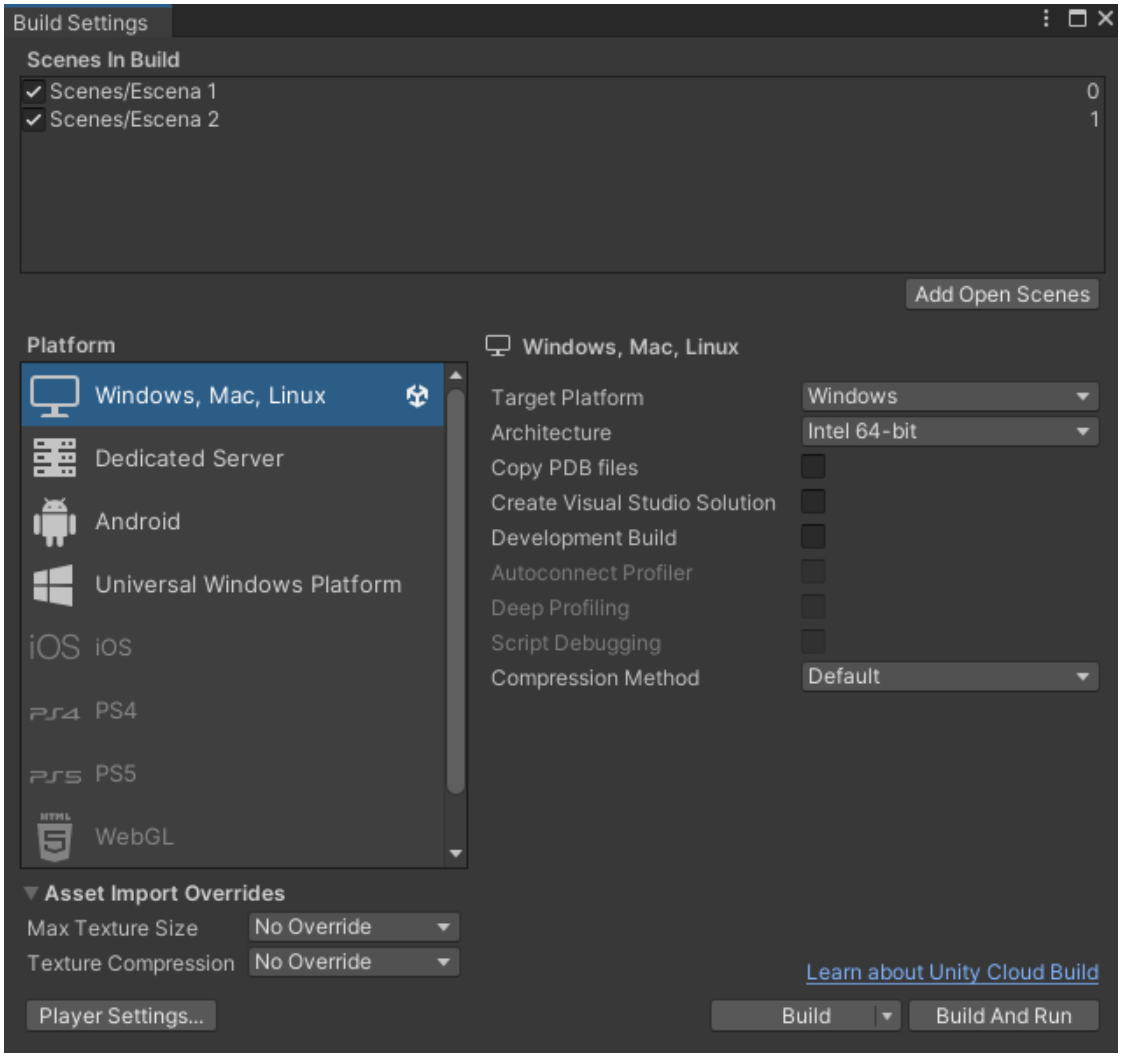

Figura 31: Exportar el proyecto. Fuente: Elaboración propia

En esta pantalla podemos configurar diferentes aspectos:

- En la parte superior podemos elegir las escenas que se incluyen en nuestro proyecto a la hora de exportarlo para ser jugado.
- A la izquierda, podemos ver las distintas plataformas disponibles para las que irá destinado el videojuego.

- A la derecha tenemos configuraciones que dependerán de la plataforma escogida. En el caso de Windows, Mac y Linux, podemos elegir el sistema operativo entre esos tres, indicar la arquitectura para la que está destinada, así como otras configuraciones más avanzadas como cambiar el método de compresión.
- En la esquina inferior izquierda podemos abrir las opciones de jugador.
- Por último, en la esquina inferior derecha se encuentra la opción de compilar y ejecutar el videojuego. Podemos así exportar nuestro proyecto a la carpeta que se quiera.

Bibliografía

Arroyo, D. (2022). Los gráficos del futuro: La nueva y aterradora demostración técnica del Unreal Engine 5. En *Meristation*. https://as.com/meristation/2022/05/09/noticias/1652114414_504738.html

Arsuaga, M. (2022, marzo 31). Inteligencia Artificial para la generación de gráficos en videojuegos. *Especiales*. https://bit.coit.es/inteligencia-artificial-para-la-generacion-de-graficos-en-videojuegos/

CRYENGINE | The complete solution for next generation game development by Crytek. (s. f.). CRYENGINE. Recuperado 7 de junio de 2022, de https://www.cryengine.com/

El Economista. (2021). *BIENVENIDOS AL 'METAVERSO', LA INMINENTE REVOLUCIÓN DE INTERNET*. https://s03.s3c.es/pdf/6/f/6f9d1ceaa53a9bc60245a5117440cb71_tecnologia.pdf

GameMaker. (s. f.). GameMaker. Recuperado 20 de mayo de 2022, de https://gamemaker.io/es

Godot Engine—Free and open source 2D and 3D game engine. (s. f.). Godot Engine. Recuperado 7 de junio de 2022, de https://godotengine.org/

González, C., Gracia, M. A., Sanagustín, L., & Romero, D. (2015). *TecsMedia: Análisis Motores gráficos y su aplicación en la industria*. https://www.aragon.es/documents/20127/674325/Estado%20del%20arte%20GameEngines%20y%20su%20impacto%20en%20la%20industria.pdf/db827568-09ef-e931-01e8-d293b9fca834

Gregory, J. (2009). *Game Engine Architecture*.

Make Your Own Game with RPG Maker. (s. f.). Recuperado 7 de junio de 2022, de https://www.rpgmakerweb.com//

Microsoft XNA Framework Redistributable 4.0. (s. f.). Microsoft Download Center. Recuperado 7 de junio de 2022, de https://www.microsoft.com/en-us/download/details.aspx?id=20914

Nvidia DLSS. (s. f.). NVIDIA. https://www.nvidia.com/es-es/geforce/technologies/dlss/

PICO-8 Fantasy Console. (s. f.). Recuperado 7 de junio de 2022, de https://www.lexaloffle.com/pico-8.php

Raya, A. (2022, marzo 22). *Esta persona no es real, ha sido creada con el motor gráfico que usarán tus próximos juegos*. https://www.eleconomista.es/tecnologia/noticias/11678666/03/22/Esta-persona-no-es-real-ha-sido-creada-con-el-motor-grafico-que-usaran-tus-proximos-juegos.html

Stencyl: Make iPhone, iPad, Android & Flash Games without code. (s. f.). Recuperado 7 de junio de 2022, de https://www.stencyl.com/

Torque3D. (s. f.). Torque3D. Recuperado 7 de junio de 2022, de https://torque3d.org

Unity Asset Store. (s. f.). Recuperado 6 de junio de 2022, de https://assetstore.unity.com/

Unity Real-Time Development Platform | 3D, 2D VR & AR Engine. (s. f.). Recuperado 7 de junio de 2022, de https://unity.com/

Unity Technologies. (s. f.). *Manual de Unity*. Unity3d.com. https://docs.unity3d.com/es/530/Manual/UnityManual.html

Unreal Engine | The most powerful real-time 3D creation tool. (s. f.). Unreal Engine. Recuperado 7 de junio de 2022, de https://www.unrealengine.com/en-US

Vallejo, D., & Cleto, M. (2015). *Desarrollo de videojuegos: Un enfoque práctico*. Los autores.

Valve Corporation. (s. f.). Valve Corporation. Recuperado 7 de junio de 2022, de https://www.valvesoftware.com/es/

Varonas, N., Pardo, L., Palazzesi, A., & Ferzzola. (2012, noviembre). *Los motores gráficos más importantes de la historia (I)*. Neoteo.com. https://www.neoteo.com/los-motores-graficos-mas-importantes-de-la-histori/

Villagrán, I. (2016). Retrospectiva de los primeros sistemas de gráficos por ordenador. *i+Diseño Rev. cient.-acad. int. Innov. Investig. Desarro. Diseño*, *11*, 34.

Villalobos, J. M. (2022, febrero 2). *Unreal Engine 5, cuando los motores de videojuegos pueden marcar el futuro del Cine*. MeriStation. https://as.com/meristation/2022/02/02/reportajes/1643795471_317184.html

Leyes referenciadas:

Real Decreto 450/2010, de 16 de abril, por el que se establece el título de Técnico Superior en Desarrollo de Aplicaciones Multiplataforma y se fijan sus enseñanzas mínimas. Boletín Oficial del Estado, 123, jueves 20 de mayo de 2010. Recuperado de https://www.boe.es/boe/dias/2010/05/20/pdfs/BOE-A-2010-8067.pdf. Última visita 20/05/2022.

Orden de 16 de junio de 2011, por la que se desarrolla el currículo correspondiente al título de Técnica Superior en Desarrollo de Aplicaciones Multiplataforma. BOJA, 142, Sevilla 21 de julio de 2011. Recuperado de https://www.juntadeandalucia.es/boja/2011/142/20. Última visita 20/05/2022.

Orden de 1 de febrero de 1996 por la que se aprueban los temarios que han de regir en los procedimientos de ingreso, adquisición de nuevas especialidades y movilidad para determinadas especialidades de los Cuerpos de Profesores de Enseñanza Secundaria y Profesores Técnicos de Formación Profesional. Recuperado de https://www.boe.es/boe/dias/1996/02/13/pdfs/C00001-00096.pdf. Última visita 24/05/2022.