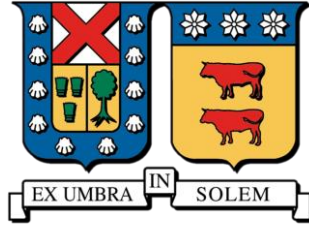


UNIVERSIDAD TÉCNICA FEDERICO SANTA MARIA
DEPARTAMENTO DE ELECTRÓNICA E INFORMÁTICA
HUALPEN - CHILE



**SISTEMA PARA CREAR NIVELES DE UN VIDEOJUEGO DE ROL-ACCIÓN EN 2D USANDO
UNITY**

RICARDO ANTONIO REVILLOUD GONZALEZ

Memoria para optar al Título de
TÉCNICO UNIVERSITARIO EN INFORMÁTICA

PROFESOR GUÍA: JORGE URRUTIA D.

DICIEMBRE 2022

RESUMEN

En este informe se examinará el **documento de diseño del juego** presentado por UwO Studios y las necesidades de los integrantes de su equipo para definir las estructuras necesarias para mantener un proyecto de un videojuego hecho en Unity. Se programan las mecánicas del juego, y herramientas que permitan a **usuarios entrenados** crear niveles para el videojuego que permitan a los jugadores jugar una partida del juego, de inicio a fin, a lo largo de una o varias sesiones de juego.

Este proyecto tendrá un enfoque Orientado a Objetos y seguirá las instrucciones de Microsoft para escribir código en C# y los consejos para estructurar proyectos de Unity Technologies.

Keywords: Unity, C#, Videojuego, 2D

ABSTRACT

This report examines the information displayed in the **game design document** provided by UwO Studios and their team member's needs to define the necessary structures to support a videogame made with Unity. The game's mechanics, and tools will be programmed to allow **trained users** to create levels for the videogame so that a player can play the game, from beginning to end, in one or multiple game sessions.

This project will have an Object Oriented focus and will follow Microsoft guidelines for writing C# code, and Unity Technologies's advices for structuring Unity projects.

GLOSARIO

Script: Secuencia de instrucciones que pueden ser ejecutadas por otro programa. En este caso Unity.

[Unity] Componente: Unidades que controlan el comportamiento de un “GameObject” dentro del videojuego, pueden ser creados por un programador utilizando MonoBehaviour..

[Unity] "GameObject": Es una representación de cualquier entidad que pueda existir dentro del mundo de un videojuego desarrollado con Unity. Es un contenedor de componentes.

[Unity] “ScriptableObject”: Esta clase permite crear contenedores de datos que pueden ser creados, visualizados, modificados y eliminados a través de la interfaz del editor de Unity.

[Unity] “MonoBehaviour”: Es la interfaz que permite escribir código que pueda ser ejecutado por los Componentes de un "GameObject".

ÍNDICE

RESUMEN	iii
ABSTRACT	iv
GLOSARIO	v
ÍNDICE	vi
INTRODUCCIÓN	10
CAPÍTULO 1: LEVANTAMIENTO DE REQUISITOS	11
PROBLEMA	11
OBJETIVO GENERAL	11
OBJETIVOS ESPECÍFICOS	11
DOCUMENTO DE DISEÑO	12
REQUISITOS FUNCIONALES	12
REQUISITOS NO FUNCIONALES	12
CAPÍTULO 2: DISEÑO DE SOFTWARE	14
MODELO ENTIDAD-RELACIÓN	15
DIAGRAMAS DE ESTRUCTURA	16
DIAGRAMA DE CLASES	16
DIAGRAMAS DE COMPORTAMIENTO	17
DIAGRAMA DE CASOS DE USO	17
DCU: DESARROLLADOR interactuando con HERRAMIENTA PROPUESTA	18
CASO: CREAR PERSONAJE JUGADOR	19
CASO: CREAR ENEMIGO	20
CASO: CREAR OBJETO	21
CASO: CREAR CONTENEDOR	22
CASO: CREA CIUDADANO	23
CASO: CREAR INTERFAZ DE USUARIO	24
	vi

CASO: CREACIÓN DE UN TRANSPORTADOR	25
DCU : JUGADOR interactuando con VIDEOJUEGO	26
CASO: CREACIÓN DE PARTIDA	27
CASO: ELIMINACIÓN DE PARTIDA	28
CASO: CONFIGURACIÓN DEL CLIENTE	29
CASO: JUGAR PARTIDA	30
CASO: DESPLAZAMIENTO DE PERSONAJE	31
CASO: INTERACCIÓN CON EL MUNDO	32
CASO: COMBATE CON ENEMIGOS	33
CASO: RECOLECCIÓN DE OBJETOS	34
CASO: DIÁLOGO CON CIUDADANOS	35
CAPÍTULO 3: HERRAMIENTAS DE DESARROLLO UTILIZADAS	36
PREPARACIÓN DEL ENTORNO DE DESARROLLO : PROGRAMAS NECESARIOS	36
UNITY	36
UNITY HUB	36
DESCARGAR UNITY HUB	36
INSTALAR UNITY HUB	37
UNITY EDITOR	37
DESCARGAR UNITY EDITOR	37
UNITY: CREANDO UN NUEVO PROYECTO	38
UNITY EDITOR: PREPARACIÓN DEL EDITOR	40
UNITY EDITOR: PREPARACIÓN DEL PROYECTO	41
DEPENDENCIAS NECESARIAS	43
A* PATHFINDING PROJECT	43
DESCARGAR A* PATHFINDING PROJECT	43
INSTALAR A* PATHFINDING PROJECT	45

CAPITULO 4: SOLUCION PROPUESTA	47
ESTRUCTURA DE ARCHIVOS	47
IMPLEMENTACIÓN DE LOS REQUERIMIENTOS	48
CREAR NUEVOS OBJETOS QUE SE PUEDAN USAR EN EL MUNDO	48
CREAR ATAQUES QUE PUEDA USAR EL JUGADOR	49
CREAR ATAQUES QUE PUEDAN USAR LOS ENEMIGOS	49
PERSONAJES	50
CREAR NUEVOS PERSONAJES ENEMIGOS	50
CREAR UN PERSONAJE PARA EL JUGADOR	51
INTERACTUABLES	51
CREAR CONTENEDORES DE OBJETOS	52
CREAR CIUDADANOS DEL MUNDO	53
CREAR TRANSPORTADORES	54
MENUS	55
CREAR EL MENÚ PRINCIPAL	55
CREAR EL MENÚ EN PARTIDA	55
INTERFACES DE JUGADOR	56
CAPÍTULO 5: COMPILACIÓN A PLATAFORMAS SELECCIONADAS	57
COMPILACIÓN A WINDOWS	57
DISTRIBUCIÓN EN LÍNEA	58
CONCLUSIONES	61
REFERENCIAS	62
ANEXOS	63
A. DOCUMENTO DE DISEÑO DEL JUEGO	63
VISIÓN GENERAL	63
OBJETIVO DEL JUEGO	63

GÉNERO	63
AUDIENCIA	63
PLATAFORMAS	64
HISTORIA	64
JUGABILIDAD	64
COMBATE	64
INTERACCIONES	65
MOVIMIENTO	66
INVENTARIO	66
EXPERIENCIA DE USUARIO (UX)	66
INTEFACES DE USUARIO	68
MENÚ PRINCIPAL, CON UNA PARTIDA GUARDADA	68
MENÚ EN PARTIDA	68
ESTADÍSTICAS DEL JUGADOR	68
INVENTARIO DEL JUGADOR	69
ESTADÍSTICAS DE LOS ENEMIGOS	69
INTERFAZ DE INTERACCIÓN	69
INTERFAZ DE DIÁLOGO	70

INTRODUCCIÓN

El presente trabajo, originado en requerimientos de la empresa UwO Studios, consistió en aplicar los conocimientos adquiridos a lo largo del proceso académico para elaborar un paquete de Unity que permita a los usuarios en UwO Studios generar niveles para un videojuego en Unity siguiendo un documento de diseño de videojuegos. Estos mundos podrían proveer contextos, tales como: educativos, investigativos, de entretenimiento, entre otros, en donde los jugadores toman control de un personaje y se les permite interactuar con un mundo y las entidades en él.

El **primer capítulo** se refiere al levantamiento de requisitos de las funciones del software.

El **segundo capítulo** incluye la presentación de los diagramas relación-entidad, actividad y comportamientos que responden a los requerimientos definidos.

El **tercer capítulo** familiariza al lector con las herramientas para hacer uso de la solución.

El **cuarto capítulo** presenta la implementación desarrollada para dar una solución a los principales requerimientos del sistema: organizar los elementos del sistema, permitir a un desarrollador crear un mundo virtual, y a un jugador interactuar con este.

El **capítulo quinto**, y final, se mostrará cómo compilar la aplicación para las plataformas tecnológicas seleccionadas.

CAPÍTULO 1: LEVANTAMIENTO DE REQUISITOS

PROBLEMA

La desarrolladora UwO Studios requiere crear un videojuego basado en el documento de diseño de su nuevo proyecto. Para esto requieren una implementación que permita a los desarrolladores organizar todos los elementos (archivos) del proyecto dependiendo de su función dentro del sistema, crear y parametrizar todas las entidades y componentes que forman el videojuego a través la interfaz del editor de Unity, sin necesidad de tener que escribir o editar código.

OBJETIVO GENERAL

Crear una estructura de archivos para el proyecto de UwO Studios y una serie de "scripts" de Unity que generen todos los comportamientos y componentes necesarios para crear el videojuego de acción y aventura que la desarrolladora quiere. Para esto, primero se extraerán los requisitos del sistema, se diseñarán los diagramas de estructura y comportamientos (desarrollador-sistema, usuario-sistema) más importantes del sistema, se diseñarán las interfaces de los componentes del editor de Unity, se documentará el proceso de instalación tanto de herramientas externas, como de paquetes internos del editor de Unity requeridos por el desarrollador, y luego se creará una escena jugable donde se puedan probar todas las mecánicas del jugador en un entorno sin errores.

OBJETIVOS ESPECÍFICOS

- A. Analizar los requerimientos del documento de desarrollo del juego.
- B. Diseñar diagramas que representan las interacciones entre el desarrollador y el sistema, y entre el jugador y el sistema.
- C. Diseñar interfaces intuitivas para facilitar el manejo de la herramienta al equipo de desarrollo.
- D. Documentar la instalación de herramientas requeridas para utilizar la solución.
- E. Construir un prototipo funcional del sistema.

DOCUMENTO DE DISEÑO

La desarrolladora provee un documento de diseño adjuntado como anexo en el presente proyecto. De ese documento podemos extraer los principales requerimientos funcionales y no funcionales para que el sistema funcione como se requiere.

REQUISITOS FUNCIONALES

- A. Los desarrolladores deben poder crear los datos de nuevos objetos y enemigos desde el editor de Unity.
- B. Los personajes deben poder desplazarse por el mapa en dos ejes.
- C. Los personajes deben poder atacar a otros personajes.
- D. Los personajes deben poder ser atacados por otros personajes.
- E. La imagen que representa al personaje debe poder girar en su eje X.
- F. Los personajes deben poder ser derrotados.
- G. El jugador debe poder controlar los movimientos de su personaje.
- H. El jugador debe poder hacer que su personaje esquive.
- I. El jugador debe poder cambiar el ataque de su personaje.
- J. El jugador debe poder atacar a otros personajes.
- K. El jugador debe poder interactuar con el objeto o entidad disponible más cercano a su posición.
- L. El jugador debe poder ver y tomar el contenido de contenedores de objetos en el mapa.
- M. El jugador debe poder dialogar con otros personajes.
- N. El jugador debe poder adquirir experiencia al derrotar enemigos.
- O. El jugador debe poder subir de nivel al acumular cierta cantidad de puntos de experiencia.
- P. Los personajes enemigos deben poder patrullar una serie de puntos.
- Q. Los personajes enemigos deben poder buscar la presencia del PJ dentro de un área a su alrededor.
- R. Los enemigos deben poder perseguir al jugador al verlo.
- S. Los enemigos deben dejar caer un contenedor con objetos al ser derrotados.

REQUISITOS NO FUNCIONALES

- A. La implementación debe ser desarrollada para el motor Unity.
- B. Los objetos y entidades del mundo deben ser representados por sprites (imágenes).
- C. El estilo artístico del mapa, objetos, entidades e interfaces debe ser consistente.
- D. El nivel de los personajes afecta su comportamiento.

- E. El personaje del jugador debe poder incrementar su efectividad a lo largo de sus partidas.
- F. El sistema debe almacenar la posición y estado de todas las entidades del mundo entre partidas.
- G. Los parámetros que controlan los comportamientos de los objetos y entidades deben poder ser manipulados desde la UI de Unity.

CAPÍTULO 2: DISEÑO DE SOFTWARE

Se extraen los requerimientos del **documento de diseño** y se crean **diagramas de estructura** y **comportamiento** que representen el sistema propuesto. Se presentará primero el diagrama que represente todos los objetos requeridos por el sistema para funcionar como se describe y posteriormente los diagramas que representan las interacciones del desarrollador con el sistema y finalmente la interacción del usuario final con el sistema.

MODELO ENTIDAD-RELACIÓN

1

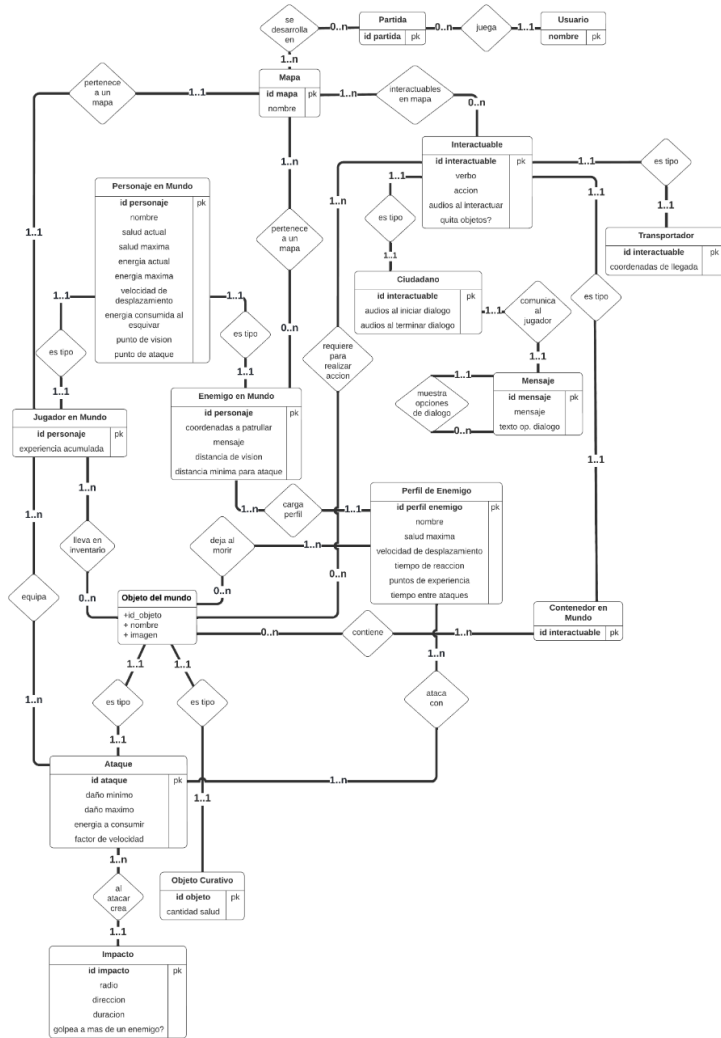


Diagrama de Entidad-Relación

Propiedades y acciones de todos los objetos, entidades y conceptos en el sistema propuesto.

¹ Ilustración 2.1: Diagrama basado en el modelo propuesto por Peter Chen en **The Entity-Relationship Model Toward a Unified View of Data**.

DIAGRAMAS DE ESTRUCTURA

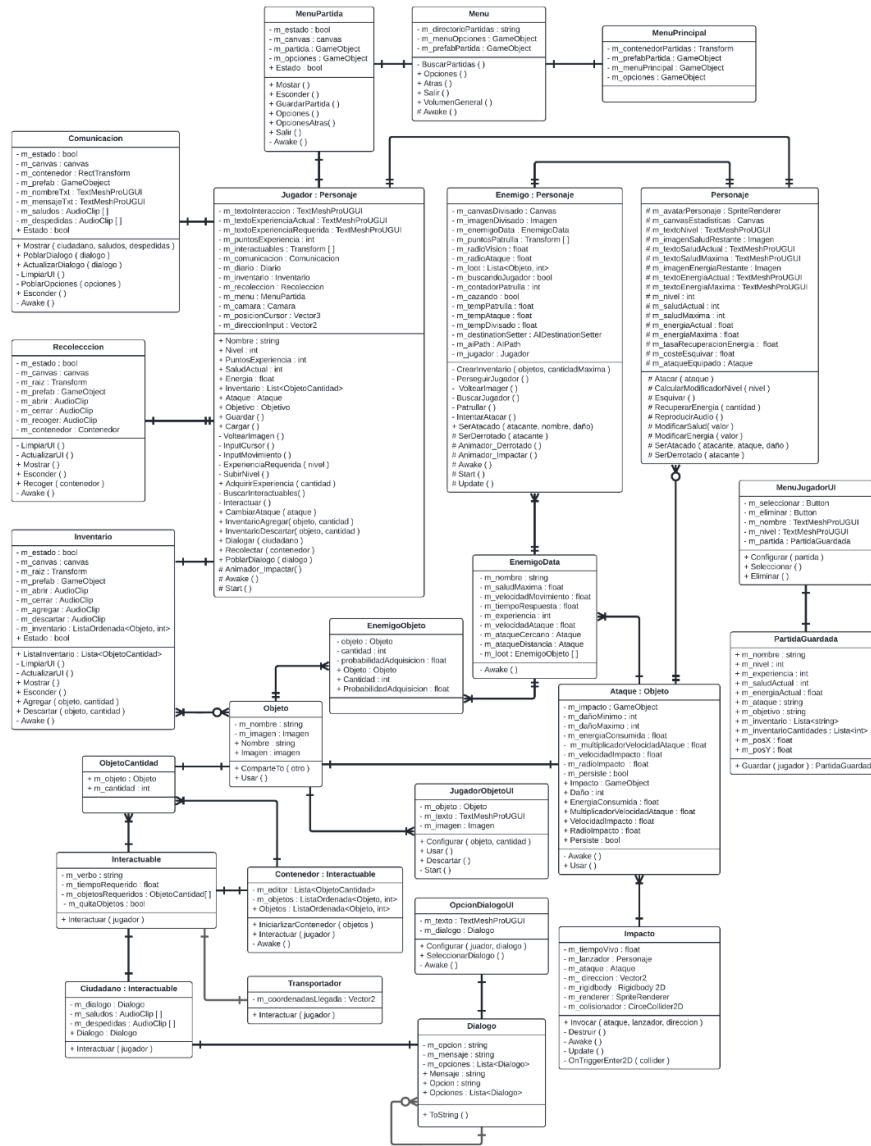
DIAGRAMA DE CLASES

Se presenta el diagrama de clases que sostiene la estructura del sistema completo, tanto para el desarrollador como para el jugador.

Diagrama de clases

Propiedades y acciones de todos los objetos, entidades y conceptos en el sistema propuesto

DIAGRAMAS DE



COMPORTAMIENTO

DIAGRAMA DE CASOS DE USO

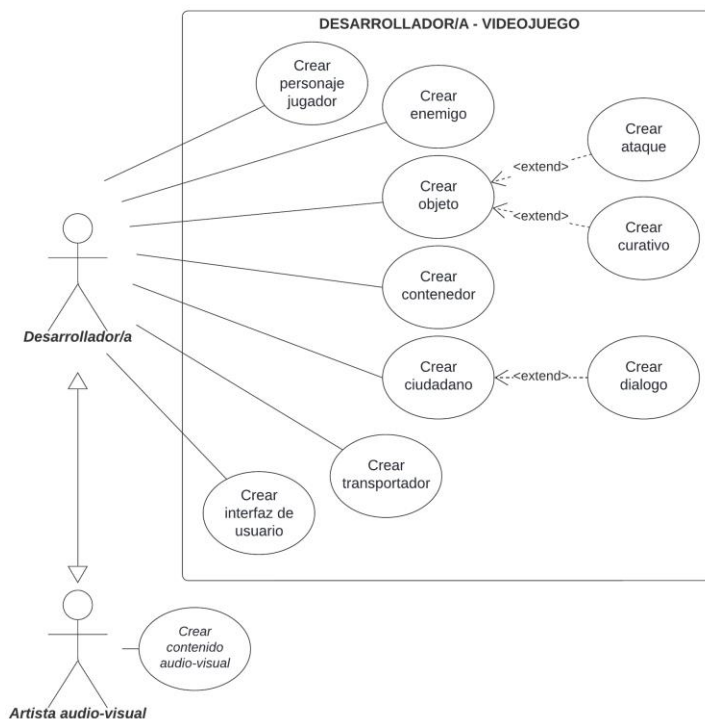
A continuación, se presentarán dos diagramas que describan los requerimientos funcionales del sistema en forma de casos de uso: el primero corresponde a las acciones que realiza el desarrollador con el sistema

² Ilustración 2.2: Diagrama basado en los información entregada por IBM para la creación de Diagramas de Clase en un curso introductorio a UML.

para crear un videojuego, y el segundo corresponde a las acciones que puede realizar el usuario final o jugador mientras experimenta el videojuego.

DCU: DESARROLLADOR interactuando con HERRAMIENTA PROPUESTA

El desarrollador interactúa con el sistema propuesto a través de la interfaz de usuario del editor de Unity, ya sea agregando Componentes a sus “*GameObjects*”, o creando “*ScriptableObjects*”.



3

Diagrama de Casos de Uso

Acciones del **desarrollador** sobre el sistema propuesto.

³ Ilustración 2.3: Acciones del desarrollador sobre el sistema propuesto. (Fuente: elaboración personal).

CASO: CREAR PERSONAJE JUGADOR

Actor(es)	Desarrollador
Descripción	El desarrollador agrega un personaje al mundo que podrá ser controlado durante la ejecución del programa.
Precondición	Hay un "GameObject" en la escena.
Postcondición	El mundo tendrá un personaje que el jugador podrá controlar.

FLUJO NORMAL PARA CREAR PERSONAJE JUGADOR

1. El **desarrollador** selecciona el "GameObject" que representará al personaje del jugador.
2. El **sistema** muestra las propiedades del "GameObject".
3. El **desarrollador** agrega el componente Jugador al "GameObject".
4. El **sistema** carga los componentes adicionales requeridos por el "script" Jugador y muestra las propiedades del componente Jugador.
5. El **desarrollador** edita las propiedades del componente del comportamiento del objeto Jugador.
6. El **sistema** guarda las propiedades y las utiliza en ejecución.

CASO: CREAR ENEMIGO

Actor(es)	Desarrollador
Descripción	El desarrollador crea y configura las estadísticas de un nuevo enemigo que podrá ser utilizado juego para combatir con el jugador.
Precondición	Hay un "GameObject" en la escena.
Postcondición	Existirá un enemigo en el mundo que reaccionará al Jugador.

FLUJO NORMAL CREAR UN ENEMIGO

1. El **desarrollador** abre el menú de creación de "assets".
2. El **sistema** muestra los tipos de "assets" que se pueden crear.
3. El **desarrollador** selecciona el tipo EnemyData.
4. El **sistema** crea un nuevo enemigo en la carpeta del **Proyecto** y muestra sus propiedades en el **Inspector**.
5. El **desarrollador** selecciona el "GameObject" que controlará la forma física del enemigo.
6. El **sistema** muestra las propiedades del "GameObject" en el **Inspector**.
7. El **desarrollador** agrega el componente Enemy al "GameObject".
8. El **sistema** carga los componentes adicionales requeridos por el "script" Enemy.
9. El **desarrollador** configura las propiedades del comportamiento del Enemy y le asigna un Enemy Data.
10. El **sistema** guarda las propiedades y los utiliza en ejecución.

FLUJO ALTERNO AL CREAR UN ENEMIGO

9. El **desarrollador** no asigna una propiedad EnemyData.
10. El **sistema**, al ejecutar el programa, inhabilita el "GameObject" al que se asignó el componente Enemy e indica el evento en la **consola**.

CASO: CREAR OBJETO

Actor(es)	Desarrollador
Descripción	El desarrollador crea y configura las estadísticas de un nuevo objeto que podrá ser utilizado juego.
Precondición	Los "scripts" se compilaron sin errores.
Postcondición	Existirá un nuevo objeto que se podrá utilizar en todos los sistemas del juego.

FLUJO NORMAL PARA CREAR OBJETO

1. El **desarrollador** abre el menú de creación de "assets".
2. El **sistema** muestra los tipos de "assets" que se pueden crear.
3. El **desarrollador** selecciona un tipo de objeto.
4. El **sistema** crea un nuevo objeto del tipo seleccionado en la carpeta del **proyecto** y muestra sus propiedades en el **inspector**.
5. El **desarrollador** configura las propiedades del nuevo objeto.
6. El **sistema** permite utilizar el objeto creado en todos los sistemas del juego.

CASO: CREAR CONTENEDOR

Actor(es)	Desarrollador
Descripción	El desarrollador crea un Contenedor lleno de objetos que el personaje del jugador puede mover a su inventario.
Precondición	Deben existir objetos que agregar al contenedor.
Postcondición	Existirá un contenedor en el mundo del que el jugador podrá recoger objetos.

FLUJO NORMAL PARA CREAR CONTENEDOR

1. El **desarrollador** selecciona un "GameObject" que quiera convertir en un contenedor.
2. El **sistema** muestra las propiedades del "GameObject" en el inspector.
3. El **desarrollador** agrega el componente Contenedor al "GameObject".
4. El **sistema** carga los componentes adicionales requeridos por el "script" Contenedor.
5. El **desarrollador** ingresa objetos dentro del contenedor.
6. El **sistema** muestra su contenido al jugador si interactúan.

FLUJO ALTERNO PARA CREAR CONTENEDOR

5. El **desarrollador** no ingresa objetos al contenedor.
6. El **sistema**, al ejecutar el programa, inhabilita el "GameObject" al que se asignó el componente Contenedor y registra el evento en la **consola**.

CASO: CREA CIUDADANO

Actor(es)	Desarrollador
Descripción	El desarrollador crea un Ciudadano con el que el personaje del jugador podrá interactuar para recibir información.
Precondición	Hay un "GameObject" en la escena.
Postcondición	El jugador podrá interactuar con un nuevo ciudadano.

FLUJO NORMAL PARA CREAR UN CIUDADANO

1. El **desarrollador** selecciona el "GameObject" que quiere designar como Ciudadano.
2. El **sistema** muestra las propiedades del "GameObject" en el inspector.
3. El **desarrollador** agrega el componente Ciudadano al "GameObject".
4. El **sistema** carga el componente Ciudadano y sus dependencias/requerimientos en el "GameObject".
5. El **desarrollador** personaliza el nombre, saludo y contenido del diálogo con el ciudadano.
6. El **sistema** muestra los diálogos del ciudadano al jugador si interactúan.

CASO: CREAR INTERFAZ DE USUARIO

Actor(es)	Desarrollador
Descripción	El desarrollador puede vincular los elementos de UI creados en Unity a los "scripts" correspondientes para manipular la apariencia y comportamiento de los elementos de la interfaz que verá el usuario final.
Precondición	Hay componentes que requieren referencias a la UI presentes en la escena.
Postcondición	Se podrán reflejar cambios en el sistema en la UI.

FLUJO NORMAL PARA CREAR INTERFAZ DE USUARIO

1. El **desarrollador** selecciona un "GameObject" que tenga un componente que requiera elementos UI.
2. El **sistema** muestra las propiedades del "GameObject" en el inspector.
3. El **desarrollador** asigna una referencia al "script" correspondiente.
4. El **sistema** se conecta con el elemento referenciado y puede editar sus propiedades.

CASO: CREACIÓN DE UN TRANSPORTADOR

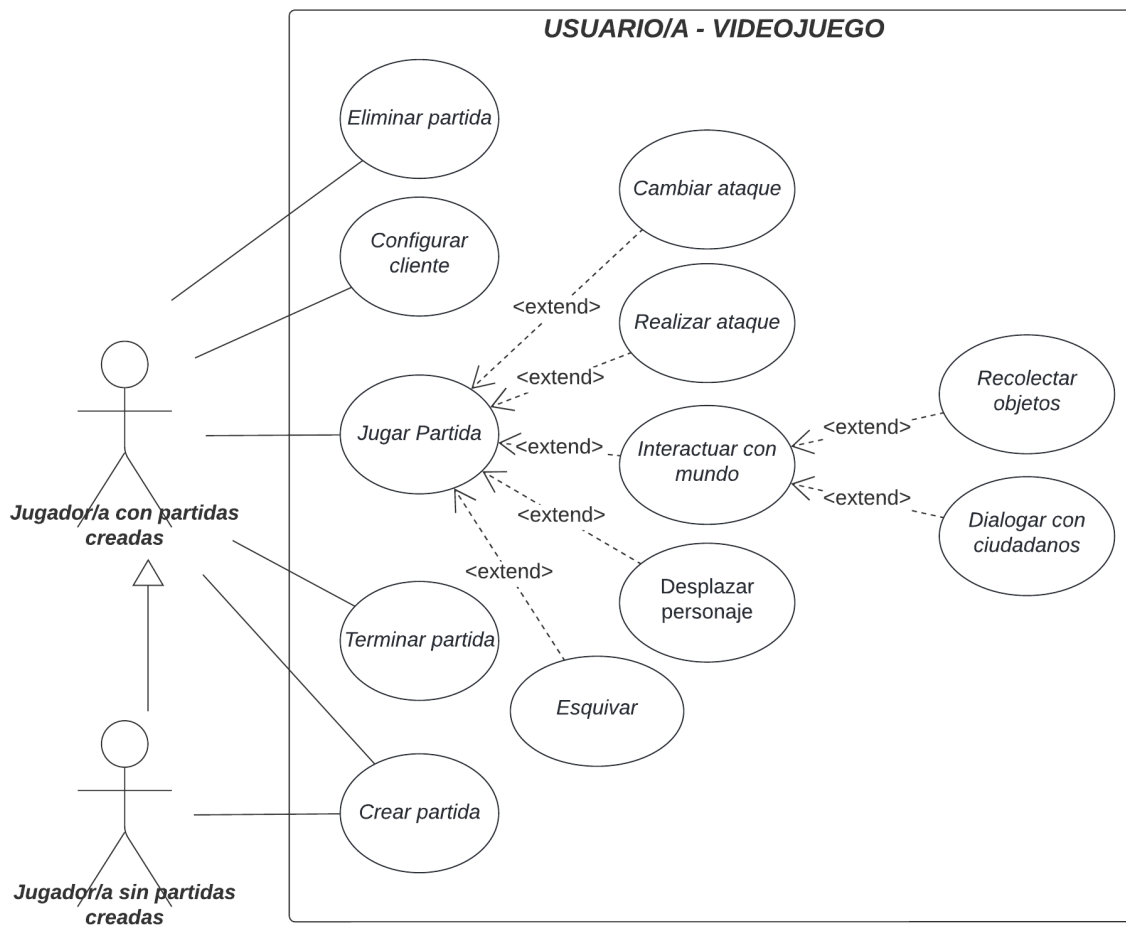
Actor(es)	Desarrollador
Descripción	El desarrollador agrega un objeto a la escena que permitirá cambiar la posición del personaje del jugador, a una indicada por el desarrollador cuando estos entren en contacto.
Precondición	Hay un "GameObject" en la escena.
Postcondición	El desarrollador podrá transportar al jugador.

FLUJO NORMAL PARA CREAR UN TRANSPORTADOR

1. El **desarrollador** agrega un componente de Ciudadano a un "GameObject".
2. El **sistema** actualiza el "GameObject" en el inspector, agrega el componente Transportador, carga sus requerimientos y valores predeterminados.
3. El **desarrollador** le da una posición de llegada al transportador.
4. El **sistema** transportará al jugador a la posición de llegada si interactúan.

DCU : JUGADOR interactuando con VIDEOJUEGO

El jugador interactúa con el sistema propuesto a través del videojuego producido por el desarrollador.



4

Diagrama de casos de uso

Acciones del **jugador** sobre el sistema propuesto

⁴ Ilustración 2.4: Acciones del jugador sobre el sistema propuesto. (Fuente: elaboración personal).

CASO: CREACIÓN DE PARTIDA

Actor(es)	Jugador sin partidas creadas, Jugador con partidas creadas
Descripción	El usuario quiere crear una nueva partida. Se crean datos que permiten iniciar una partida con un personaje nuevo.
Precondición	El jugador se encuentra en el menú principal.
Postcondición	El jugador podrá seleccionar una nueva partida para jugar.

FLUJO NORMAL PARA CREAR UNA PARTIDA

1. El **jugador** selecciona la opción para crear una nueva partida.
2. El **sistema** solicita el ingreso de un nombre para el personaje.
3. El **jugador** ingresa un nombre para su personaje.
4. El **sistema** carga el primer nivel del juego y ubica al personaje en la posición inicial.
5. El **jugador** asume el control del personaje.
6. El **sistema** guarda los datos de la nueva partida en disco.

CASO: ELIMINACIÓN DE PARTIDA

Actor(es)	Jugador con partidas creadas
Descripción	El jugador selecciona una partida creada para borrar. Se limpia la información de esa partida y el jugador puede empezar una nueva.
Precondición	El jugador se encuentra en el menú principal. Hay al menos una partida creada.
Postcondición	La partida seleccionada se eliminará y se podrá crear otra en su lugar.

FLUJO NORMAL PARA ELIMINAR UNA PARTIDA

1. El **jugador** selecciona una partida a eliminar.
2. El **sistema** solicita confirmar la eliminación del personaje seleccionado.
3. El **jugador** confirma la selección.
4. El **sistema** elimina los datos guardados.

FLUJO ALTERNO PARA ELIMINAR UNA PARTIDA

3. El **jugador** cancela la eliminación.
4. El **sistema** vuelve al menú principal.

CASO: CONFIGURACIÓN DEL CLIENTE

Actor(es)	Jugador sin partidas creadas, Jugador con partidas creadas
Descripción	El jugador configura el cliente para poder jugar de una forma cómoda.
Precondición	El volumen no es óptimo para el usuario.
Postcondición	El volumen fue cambiado a niveles óptimos para el usuario.

FLUJO NORMAL PARA CONFIGURAR EL CLIENTE

1. El **jugador** selecciona abrir las opciones.
2. El **sistema** muestra la interfaz de opciones.
3. El **usuario** ajusta las opciones.
4. El **sistema** guarda los cambios.

CASO: JUGAR PARTIDA

Actor(es)	Jugador con partidas creadas
Descripción	El jugador se encuentra controlando un personaje dentro del mundo del videojuego, y puede interactuar con éste.
Precondición	El jugador tiene una partida creada.
Postcondición	El jugador se encuentra dentro de una partida, controlando a un personaje con las estadísticas seleccionadas.

FLUJO NORMAL PARA JUGAR PARTIDA

1. El **jugador** selecciona un personaje para jugar
2. El **sistema** carga los datos del personaje y muestra lo deja en su última posición.
3. El **jugador** queda en control de su jugador y puede interactuar con el mundo.
4. El **sistema** refleja las instrucciones del jugador dentro del mundo.

FLUJO ALTERNO PARA JUGAR PARTIDA

1. El **sistema** no pudo cargar los datos del personaje.
2. El **sistema** vuelve al menú principal.
3. El **sistema** muestra un error indicando que no se pudo cargar al personaje.

CASO: DESPLAZAMIENTO DE PERSONAJE

Actor(es)	Jugador con partidas creadas
Descripción	Cuando el jugador pulsa una tecla de movimiento, su personaje se desplaza en la dirección indicada.
Precondición	El jugador se encuentra jugando una partida.
Postcondición	El personaje del jugador se ha desplazado en el mapa.

FLUJO NORMAL PARA DESPLAZAR A UN PERSONAJE

1. El **jugador** pulsa una tecla de dirección.
2. El **sistema** mueve al personaje del jugador en la dirección indicada.

FLUJO ALTERNO PARA DESPLAZAR A UN PERSONAJE

1. Si el **jugador** intenta moverse hacia un muro, no podrá atravesarlo.

CASO: INTERACCIÓN CON EL MUNDO

Actor(es)	Jugador con partidas creadas
Descripción	Cuando el jugador se acerca a un área de interacción, se muestra un mensaje indicando que interacción se puede llevar a cabo. El jugador puede pulsar una tecla para llevar a cabo la interacción y el estado del jugador o el mundo se actualizan para informar que el cambio se llevó a cabo.
Precondición	El jugador se encuentra jugando una partida
Postcondición	El personaje del jugador realizó una interacción que generó una respuesta del mundo.

FLUJO NORMAL PARA INTERACTUAR CON EL MUNDO

1. El **sistema** notifica al jugador que puede interactuar.
2. El **jugador** indica que quiere realizar la interacción.
3. El **sistema** realiza la interacción mostrada.

FLUJO ALTERNO PARA INTERACTUAR CON EL MUNDO

2. El **jugador** que no cumple los requisitos de interacción indica que quiere realizar la interacción.
3. El **sistema** no puede realizar la interacción.
4. El **sistema** muestra una notificación indicando por qué no se pudo llevar a cabo la interacción.

CASO: COMBATE CON ENEMIGOS

Actor(es)	Jugador con partidas creadas
Descripción	El jugador puede lanzar un ataque en dirección a un enemigo, si este hace contacto entonces el enemigo recibe daño a su salud.
Precondición	El jugador se encuentra jugando una partida. Un ataque del jugador hizo contacto con un enemigo
Postcondición	El personaje habrá infringido daño a un enemigo. Se habrá drenado la energía del jugador.

FLUJO NORMAL PARA COMBATIR CON ENEMIGOS

1. El **jugador** utiliza su ataque.
2. El **sistema** reduce los recursos del jugador, reproduce la animación de ataque.
3. El **jugador** apunta su ataque.
4. El **sistema** genera las colisiones del ataque en la dirección donde apunta el jugador, y reduce la salud de un enemigo si hace contacto con este.

FLUJO ALTERNO PARA COMBATIR CON ENEMIGOS

4. El **sistema** genera las colisiones del ataque, sin embargo el ataque no alcanzó a dañar a ningún enemigo.

CASO: RECOLECCIÓN DE OBJETOS

Actor(es)	Jugador con partidas creadas
Descripción	Cuando el jugador interactúa con un contenedor, este muestra su contenido en la interfaz de usuario. Si el jugador vuelve a pulsar el botón de interacción mientras la interfaz está abierta, entonces moverá todos los objetos que pueda del contenedor a su inventario.
Precondición	El jugador se encuentra jugando una partida. Un ataque del jugador hizo contacto con un enemigo
Postcondición	El personaje habrá infringido daño a un enemigo. Se habrá drenado la energía del jugador.

FLUJO NORMAL PARA RECOLECTAR OBJETOS

1. El **sistema** muestra el contenido del contenedor en la interfaz de usuario.
2. El **jugador** selecciona que quiere recoger los objetos.
3. El **sistema** mueve los objetos del contenedor al inventario del jugador.

FLUJO ALTERNO PARA RECOLECTAR OBJETOS

3. Si el jugador no tiene suficiente espacio para los objetos del contenedor en su inventario, solo tomará los que entren hasta llenar la capacidad del inventario y dejará el resto dentro del contenedor.

CASO: DIÁLOGO CON CIUDADANOS

Actor(es)	Jugador con partidas creadas
Descripción	Cuando el jugador interactúa con un ciudadano, éste muestra su saludo y las opciones de diálogo disponibles para la conversación actual.
Precondición	El jugador se encuentra jugando una partida. Un ataque del jugador hizo contacto con un enemigo
Postcondición	El personaje habrá infringido daño a un enemigo. Se habrá drenado la energía del jugador.

FLUJO NORMAL PARA DIALOGAR CON CIUDADANOS

1. El **sistema** muestra el contenido del contenedor en la interfaz de usuario.
2. El **jugador** selecciona que quiere recoger los objetos.
3. El **sistema** mueve los objetos del contenedor al inventario del jugador.

FLUJO ALTERNO PARA DIALOGAR CON CIUDADANOS

3. Si el jugador no tiene suficiente espacio para los objetos en su inventario, entonces el **sistema** solo moverá los objetos que quepan y el resto quedarán en el contenedor.

CAPÍTULO 3: HERRAMIENTAS DE DESARROLLO UTILIZADAS

PREPARACIÓN DEL ENTORNO DE DESARROLLO : PROGRAMAS NECESARIOS

UNITY

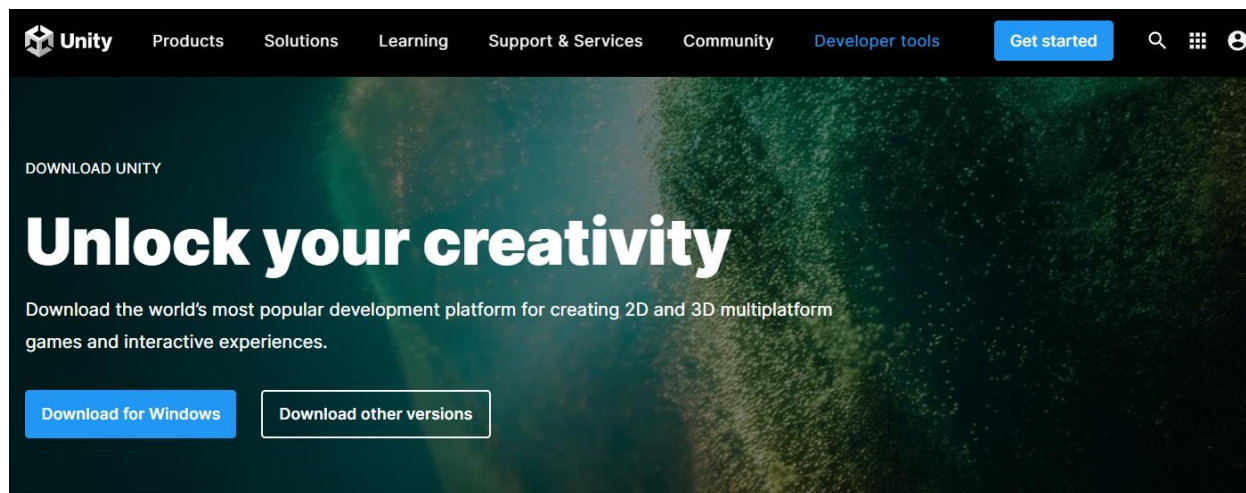
Unity es una plataforma que provee a usuarios la capacidad de ubicar recursos audiovisuales en un entorno virtual que pueda responder en tiempo real a entradas del usuario o eventos del sistema. Para trabajar con los sistemas de Unity se requiere descargar un **Editor** de Unity a través del gestor de versiones de Unity: **Unity Hub**.

UNITY HUB

Use the Unity Hub to manage multiple installations of the Unity Editor, create new projects, and access your work.

DESCARGAR UNITY HUB

UnityHub puede ser descargado navegando a la página de descargas de Unity: <https://www.unity.com/downloads>, y seleccionando la versión para el sistema operativo correspondiente.



5

Unity Hub

Página de descarga de Unity.

INSTALAR UNITY HUB

Ejecutar el instalador de Unity Hub, seleccionar el directorio de instalación, y completar normalmente la instalación.

UNITY EDITOR

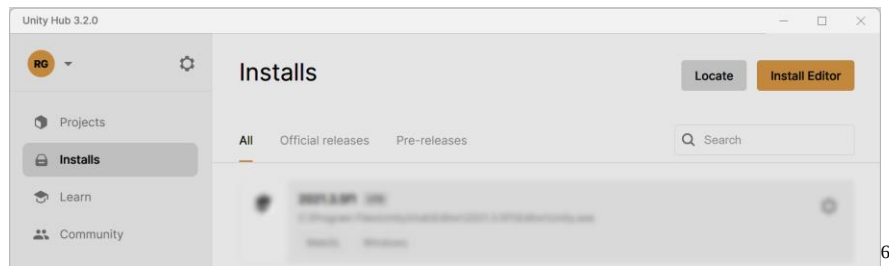
Unity Editor es el programa que permite a los usuarios crear las experiencias virtuales mediante su interfaz de usuario. Tiene opciones para personalizar el proyecto, la misma ventana del editor, y la compilación del proyecto entre otras.

DESCARGAR UNITY EDITOR

⁵ Ilustración 3.1: Página de descarga de Unity. (Fuente: Unity)

Ejecutar Unity Hub, seleccionar el botón “***Installs***” Instalaciones en el panel de navegación para ir a la ventana de instalación de editores.

En la ventana de instalaciones, seleccionar “***Install Editor***” Instalar Editor para instalar un nuevo editor.



Unity Hub

Sección de instalación de editores.

Se selecciona la versión deseada del editor y, en siguiente paso, los módulos necesarios, como por ejemplo soporte para compilar a otras plataformas en la categoría **plataformas**.

Si no se tiene un editor de código o un IDE se recomienda marcar la casilla Microsoft Visual Studio Community en la categoría **herramientas de desarrollo**, el proceso de instalación de Visual Studio comenzará una vez termine la descarga e instalación del editor.

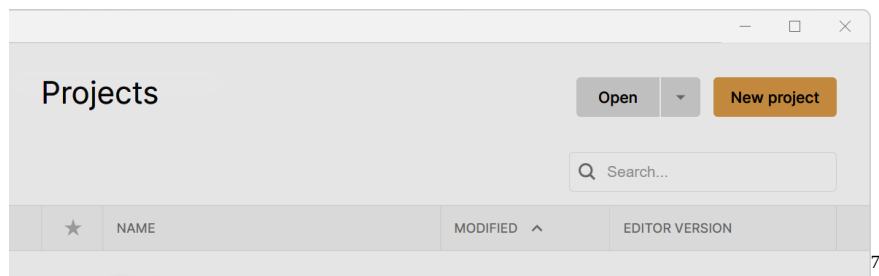
Finalmente, si se requiere se puede marcar la casilla documentación para obtener una copia “*offline*” de toda la documentación del editor en forma local.

UNITY: CREANDO UN NUEVO PROYECTO

Dentro de Unity Hub, seleccionar el botón “***Projects***” Proyectos en el menú de navegación para abrir la ventana de gestión de proyectos.

En la ventana de gestión de proyectos, seleccionar “***New project***” Nuevo proyecto para comenzar a crear un nuevo proyecto.

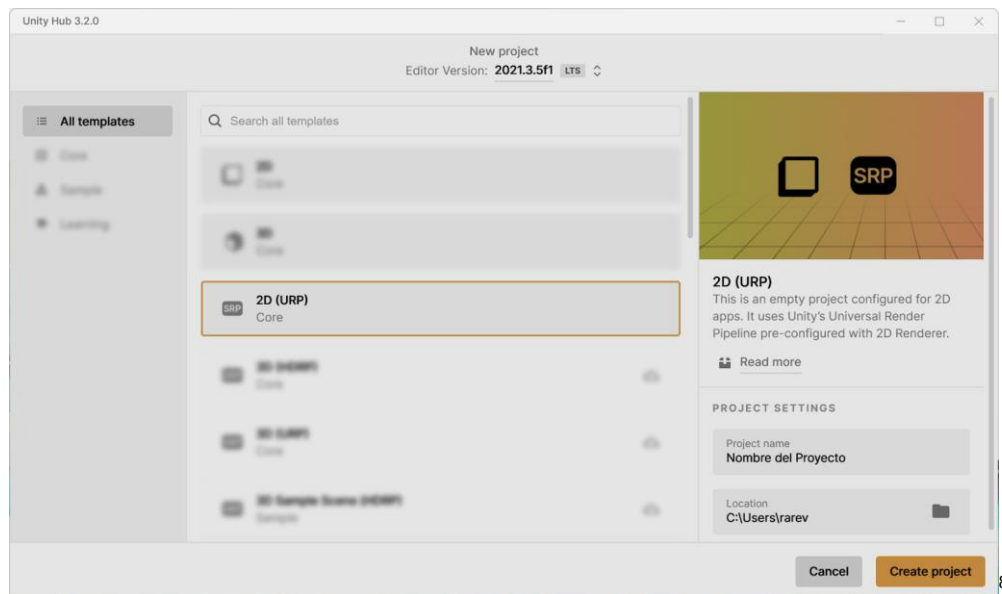
⁶ Ilustración 3.2: Ventana de instalación de Editores en Unity Hub(Fuente: Unity Hub)



Unity Hub

Creando un nuevo proyecto

En la ventana de nuevo proyecto, seleccionar la “**Editor Version**” Versión del Editor deseada. Por defecto se selecciona la más reciente, como tipo de proyecto se selecciona **2D (URP)**, y para terminar se le da un nombre al proyecto, y opcionalmente se puede cambiar el directorio de creación.



Unity Hub

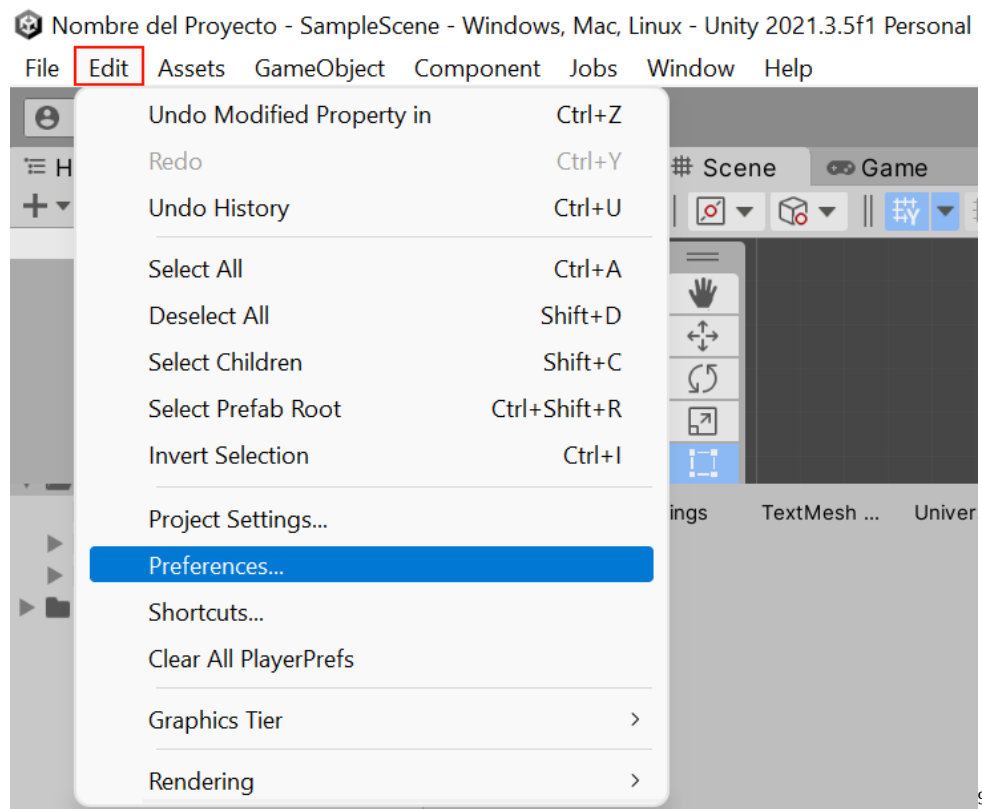
Configurando un nuevo proyecto

⁷ Ilustración 3.3: Sección de Proyectos en Unity Hub (Fuente: Unity Hub)

⁸ Ilustración 3.4: Ventana de creación de un Proyecto en Unity Hub (Fuente: Unity Hub)

UNITY EDITOR: PREPARACIÓN DEL EDITOR

Hay que indicarle al editor qué editor de texto deberá usar para abrir "scripts", esto se puede hacer en las preferencias del editor, a las que se accede seleccionando “**Edit**” Editar en la barra superior, y luego “**Preferences**” Preferencias.

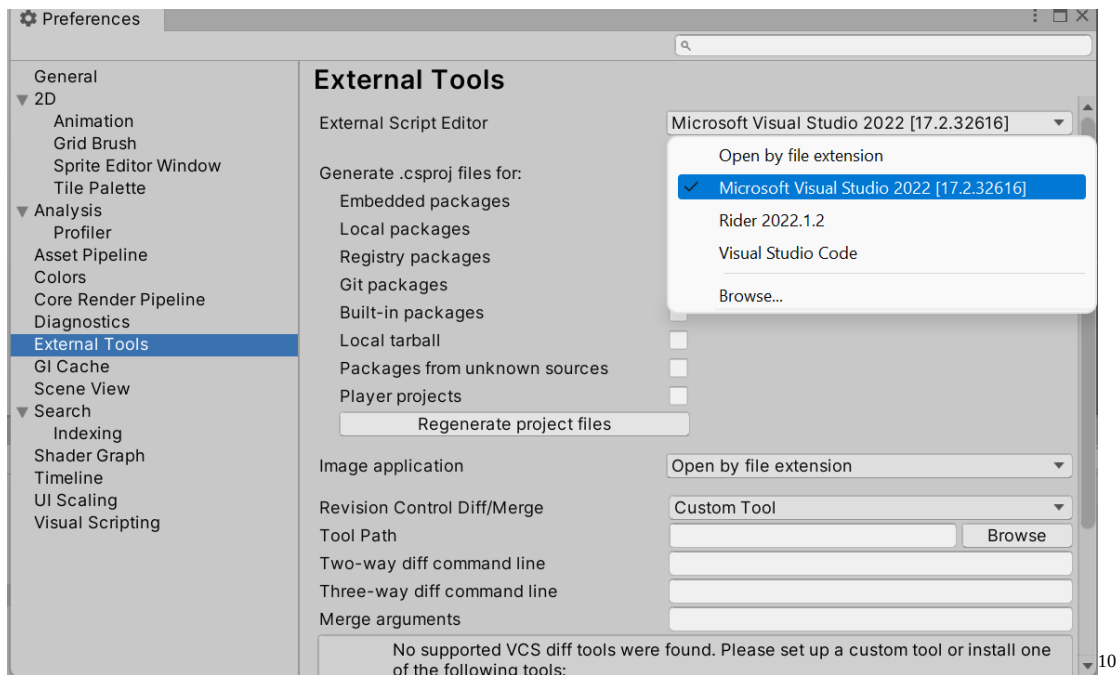


Unity Editor

Abriendo la ventana de preferencias del editor.

En la ventana de preferencias, navegar a “**External Tools**” Herramientas externas, y en “**External "script" Editor**” Editor externo de "scripts" seleccionar el editor deseado del menú desplegable.

⁹ Ilustración 3.5: Menú “Editar” de Unity. (Fuente: Unity Editor 2021.3.5f1)



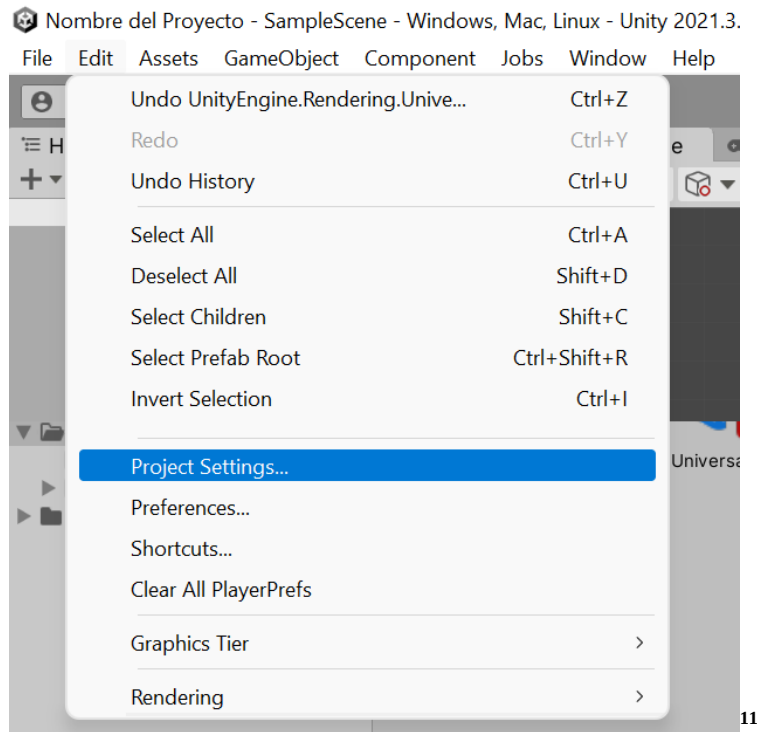
Unity Editor

Seleccionando Microsoft Visual Studio como editor externo de "scripts".

UNITY EDITOR: PREPARACIÓN DEL PROYECTO

El sistema de texto por defecto de Unity es bastante simple y de una resolución baja, además de no presentar opciones que se requieren para este sistema, para solucionar esto podemos importar un segundo sistema de texto: TextMesh Pro, un sistema de texto más pesado, pero completo desarrollado por Unity Technologies. Esto se puede hacer seleccionando *Edit* (Editar) en la barra superior, y luego *Project Settings* (Configuraciones del proyecto).

¹⁰ Ilustración 3.6: Preferencias del Editor de Unity (Fuente: Unity Editor 2021.3.5f1)

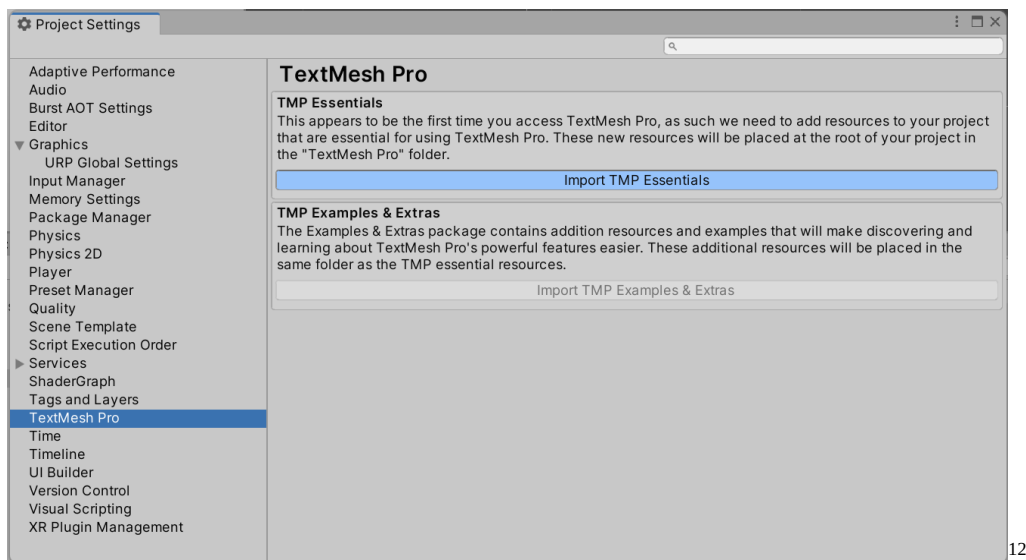


Unity Editor

Abriendo la ventana de preferencias del proyecto.

En la ventana de configuraciones del proyecto, navegar a **TextMesh Pro** y seleccionar **Import TMP Essentials** Importar elementos esenciales de TMP. Esta acción puede tomar unos minutos.

¹¹ Ilustración 3.7: Menú “Editar” de Unity (Fuente: Unity Editor 2021.3.5f1)



Unity Editor

Importando elementos esenciales de TMP

DEPENDENCIAS NECESARIAS

A* PATHFINDING PROJECT

Desarrollado por Aaron Granberg, A* Pathfinding Project es una implementación del algoritmo de búsqueda de en grafos A* como paquete de Unity que habilita, una vez importado al proyecto, la generación de grafos que permiten a la computadora navegar por un mapa, esquivando obstáculos en ambientes 2D o 3D.

DESCARGAR A* PATHFINDING PROJECT

Descargar el paquete desde la página web de Aaron Granberg: <https://arongranberg.com/astar/download>, seleccionando la versión gratuita.

¹² Ilustración 3.8: Preferencias del Proyecto de Unity (Fuente: Unity Editor 2021.3.5f1)

Download

Here you can download the package. A free version and a pro version is available as well as a tutorial. If you are interested in the differences between the free and the pro version, take a look at [this page](#). The changelog can be found [here](#).

Pro Version

If you have purchased the pro version of the package, enter the invoice number (if you bought it via email) or the email (if you have bought it on this site). After you have submitted the form, the download button for the pro version will become enabled.

Free Version

Free version of the project. Slightly more limited than the pro version, but still very powerful. Go ahead and try it out! Includes several example scenes showing how to use the system.

 Download (version: 4.2.17, last updated: 06 Nov 2021)

13

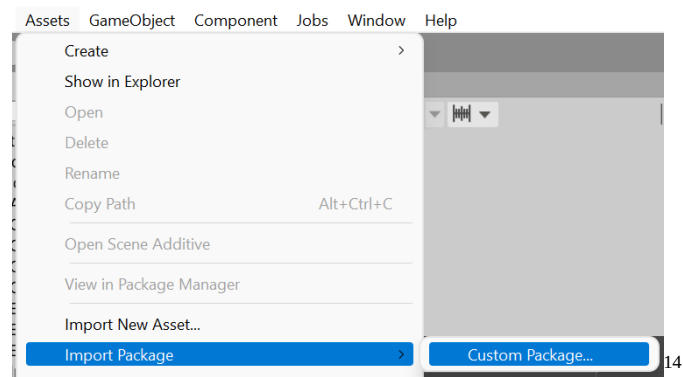
A* Pathfinding Project

Descargando la versión gratuita del proyecto

¹³ Ilustración 3.9: Página de descarga de A* Pathfinding Project, por Aron Granberg (Fuente: Aaron Granberg)

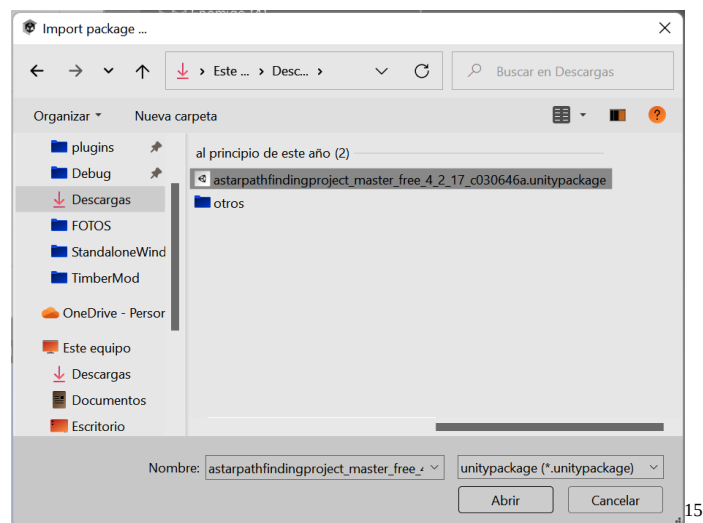
INSTALAR A* PATHFINDING PROJECT

Una vez descargado, importar el paquete a Unity navegando a **Assets** Activos en la barra superior, y seleccionando **Import Package** Importar Paquete, **Custom Package** Paquete Personalizado para abrir la ventana de selección del paquete.



Unity Editor

Importando paquetes personalizados



Unity Editor

Importando paquetes personalizados

¹⁴ Ilustración 3.10: Menú de importación de paquetes de Unity (Fuente: Unity Editor 2021.3.5f1)

¹⁵ Ilustración 3.11: Cuadro de dialogo de selección de paquetes de Unity (Fuente: Unity Editor 2021.3.5f1)

Seleccionar, y abrir el paquete descargado.

Quitar del listado los elementos que no se requieran desmarcando sus respectivas casillas, y seleccionando

Import Importar. **Se recomienda dejar todo si no se sabe lo que se hace.**

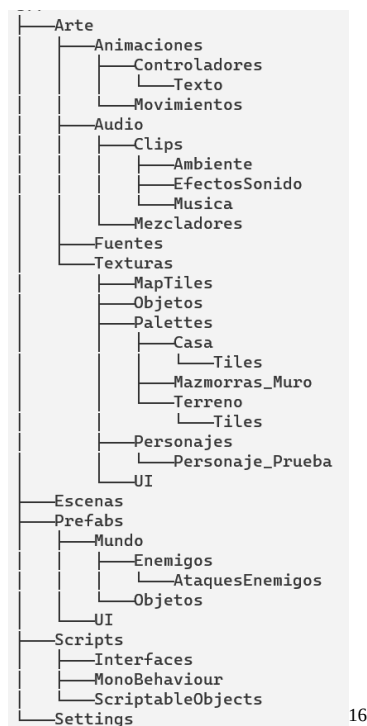
Comenzará el proceso de importar el paquete, esto podría tardar unos minutos.

CAPITULO 4: SOLUCION PROPUESTA

ESTRUCTURA DE ARCHIVOS

Por defecto un proyecto de Unity viene con dos carpetas: "Scenes" y "Settings". Scenes es una carpeta destinada a almacenar los archivos que contienen los mapas creados y "Settings" almacena archivos de configuración del proyecto. Eliminaremos la carpeta Scenes.

Para poder usar de forma más efectiva la carpeta de proyecto, se crea una estructura de directorios dentro de la carpeta Assets, que contiene todos los elementos que se pueden ver desde el explorador de proyecto.



Estructura de archivos

Estructura de los directorios del proyecto de Unity

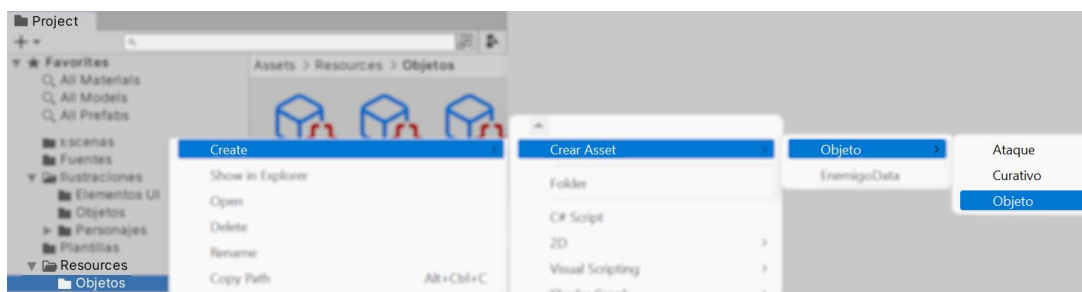
¹⁶ Ilustración 4.1: Estructura de los directorios del proyecto de Unity impresa en la consola de comandos con el comando "tree" (Fuente: Elaboración propia)

IMPLEMENTACIÓN DE LOS REQUERIMIENTOS

En la siguiente sección se mostrará cómo hacer uso de los elementos de la interfaz de usuario creados por el sistema propuesto.

CREAR NUEVOS OBJETOS QUE SE PUEDAN USAR EN EL MUNDO

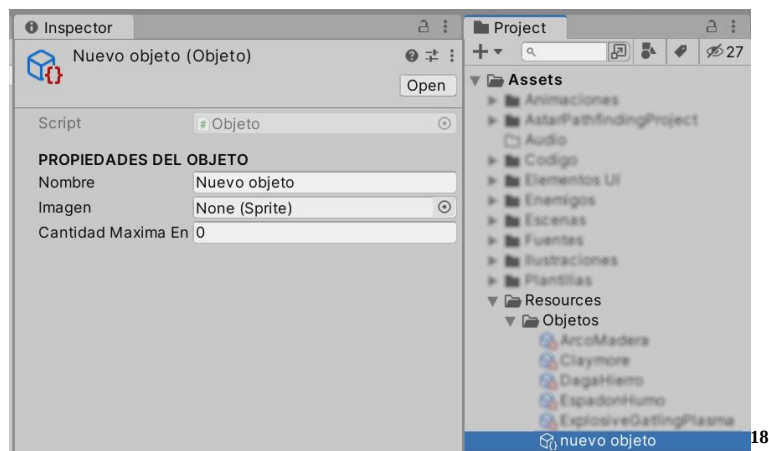
Para agregar un nuevo objeto al juego es necesario hacer clic derecho en el directorio **/Assets/Objetos** y seleccionar **“Create”/Crear “Asset”/Objeto/[Tipo de objeto]**.



Menú contextual del explorador de proyectos de Unity

Creando un nuevo objeto base.

¹⁷ Ilustración 4.3: Procedimiento para crear un nuevo objeto del juego a través de la interfaz de Unity. (Fuente: Unity Editor 2022.3.5f1)



Objeto

Propiedades de un objeto expuestas al desarrollador

Para que el jugador pueda acceder al objeto y hacer uso de este en el juego es necesario agregar el objeto a un **Contenedor** o a la lista de objetos de un **Enemigo**.

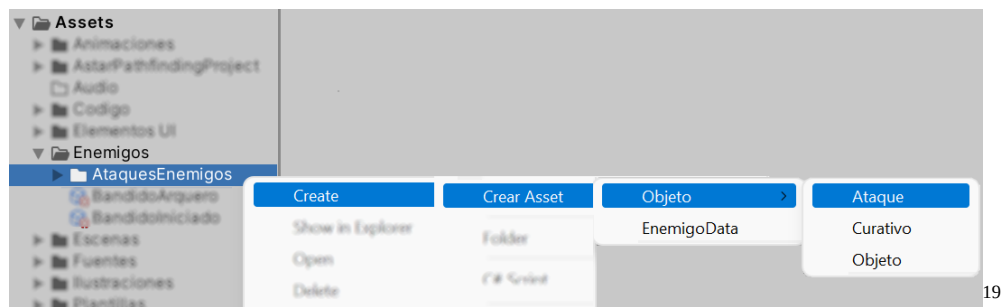
CREAR ATAQUES QUE PUEDA USAR EL JUGADOR

Para agregar un nuevo ataque que pueda usar un jugador, es necesario crearlo en el directorio /"Assets"/Resources/Objetos, haciendo clic derecho en el directorio y seleccionando "**Create**"/**Crear "Asset"/Ataque** en el menú contextual. Se creará un nuevo archivo **.asset** conteniendo la información del nuevo ataque.

CREAR ATAQUES QUE PUEDAN USAR LOS ENEMIGOS

Para crear el ataque de un enemigo el proceso es similar, pero el directorio donde se crea el asset cambia por: /"Assets"/Enemigos/AtaquesEnemigos, para mantener el rendimiento de la aplicación.

¹⁸ Ilustración 4.4: Propiedades del nuevo objeto creado. (Fuente: Unity Editor 2022.3.5f1)



Ataque

Creando un nuevo ataque que será usado por enemigos.

Una vez creado el ataque es necesario cambiar sus propiedades seleccionandolo en el explorador del proyecto y cambiando sus propiedades en el inspector.

PERSONAJES

Los “personajes” son las entidades del mundo con la capacidad de atacar y ser atacados.

CREAR NUEVOS PERSONAJES ENEMIGOS

Para crear un nuevo enemigo es necesario hacer clic derecho en el directorio **/"Assets"/Enemigos** y seleccionar **Crear/Nuevo "Asset"/EnemigoData** en el menú contextual. Se creará un nuevo archivo **.asset** en blanco que contiene la información de un enemigo, incluyendo nombre, estadísticas, y ataques.

¹⁹ Ilustración 4.5: Procedimiento para crear un nuevo ataque que será usado por enemigos. (Fuente: Unity Editor 2022.3.5f1)

Para agregar el enemigo a una escena es necesario agregar el componente **Enemigo** a un "GameObject" y en la propiedad EnemyData del componente Enemy seleccionar la información del enemigo que se quiere crear.

CREAR UN PERSONAJE PARA EL JUGADOR

El personaje jugador se puede agregar a una escena agregando el componente **Jugador** a un "GameObject", esto agrega también los requisitos indicados en el "script": **Inventario, Recolección, Comunicación y Menú Partida.**

El componente **Jugador** controla el comportamiento del personaje del jugador. Manifiesta los *inputs* del usuario dentro del programa, controlando el movimiento del personaje, sus ataques, la gestión de su inventario, la recolección de objetos, y la capacidad de comunicarse con los ciudadanos.

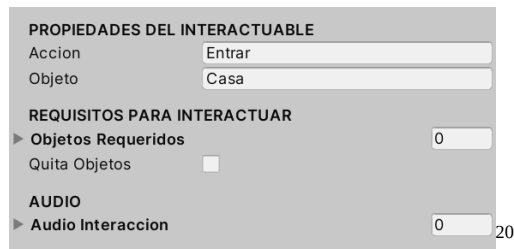
INTERACTUABLES

Los interactivables son componentes que presentan al jugador con la oportunidad de interactuar con ellos para adquirir información u objetos. Requieren indicar el nombre de la **acción** que se llevará a cabo cuando el jugador interactúe con el objeto, y el **nombre del objeto** con el que se va a interactuar.

También cuentan con un listado de objetos que podrían ser requeridos para completar la interacción, dando la opción de remover o mantener los objetos en el inventario del personaje una vez completada la interacción.

El componente Interactable **NO PUEDE SER UTILIZADO POR SÍ SOLO**, es una interfaz para uniformar otros tipos de interactivables.

Todos los componentes de tipo Interactable **REQUIEREN** también un componente de tipo Collider2D Cualquiera. Si el componente Interactable no encuentra un Collider2D al iniciar el juego, entonces este será deshabilitado y no podrá usarse.

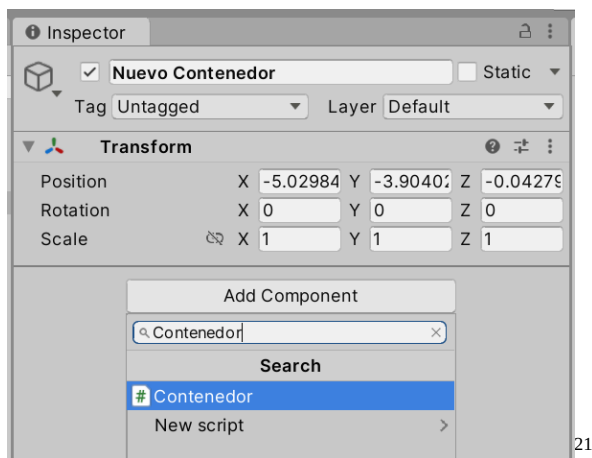


Interactable

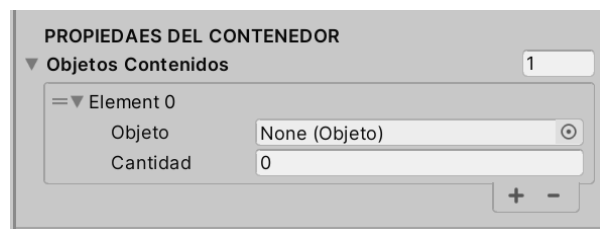
Propiedades comunes de un interactuable expuestas a un desarrollador

CREAR CONTENEDORES DE OBJETOS

Para crear un nuevo contenedor es necesario agregar el componente **Contenedor** a un "GameObject". En las propiedades del componente, junto a las propiedades propias de un interactuable, se pueden agregar objetos al listado **Objetos Contenidos** y asignarles una cantidad para que aparezcan dentro del contenedor en el juego.



Contenedor de objetos



Contenedor de objetos

²⁰ Ilustración 4.6: Propiedades de los objetos tipo Interactuables. (Fuente: Unity Editor 2022.3.5f1)

²¹ Ilustración 4.7: Panel para agregar componentes a un "GameObject" (Fuente: Unity Editor 2022.3.5f1)

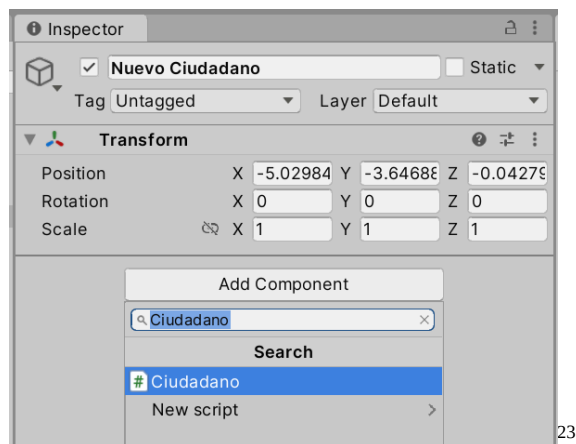
²² Ilustración 4.8: Propiedades específicas de un Interactable tipo Contenedor (Fuente: Unity Editor 2022.3.5f1)

Agregando componente **Contenedor** a un "GameObject".

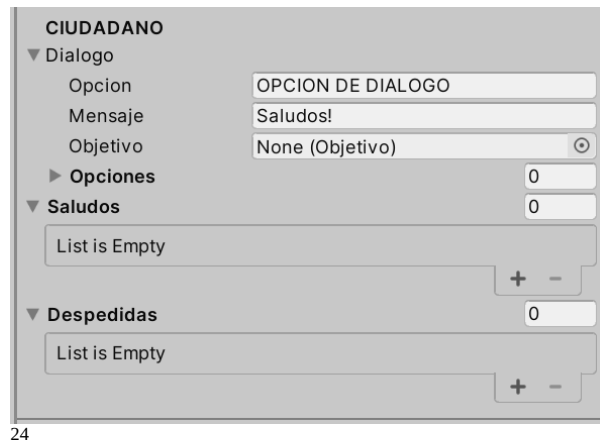
Propiedades expuestas al desarrollador en el inspector.

CREAR CIUDADANOS DEL MUNDO

Para crear una entidad con la que el jugador pueda dialogar es necesario agregar el componente **Ciudadano** a un "GameObject". En las propiedades del objeto, junto con las propiedades intrínsecas de un objeto interactuable, hay que definir las propiedades propias del ciudadano, como el mensaje con el que recibe al personaje, las opciones de diálogo que tendrá disponibles y los sonidos que hará al comenzar y terminar la conversación.



Ciudadano
Agregando un componente Ciudadano a un "GameObject".



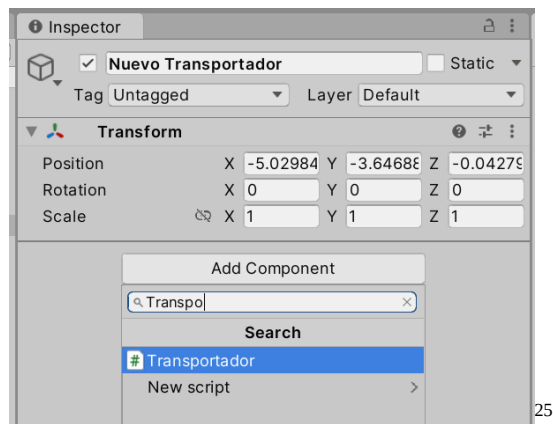
Ciudadano
Propiedades expuestas al desarrollador en el inspector.

²³ Ilustración 4.9: Panel para agregar componentes a un "GameObject" (Fuente: Unity Editor 2022.3.5f1)

²⁴ Ilustración 4.10: Propiedades de un Ciudadano (Fuente: Unity Editor 2022.3.5f1)

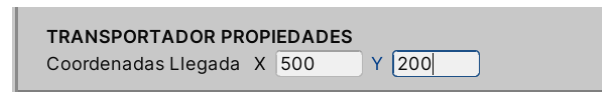
CREAR TRANSPORTADORES

Para crear un transportador que pueda cambiar la posición del jugador es necesario agregar el componente **Transportador** a un "GameObject". Junto con las propiedades comunes de un interactuable es necesario asignar las coordenadas del mundo al que se será movido el personaje del jugador cuando interactúe con el componente.



Transportador

Agregando el componente **Transportador** a un "GameObject".



Transportador

Propiedades expuestas al desarrollador en el inspector.

²⁵ Ilustración 4.11: Panel para agregar componentes a un "GameObject". (Fuente: Unity Editor 2022.3.5f1)

²⁶ Ilustración 4.12: Propiedades de un Transportador(Fuente: Unity Editor 2022.3.5f1)

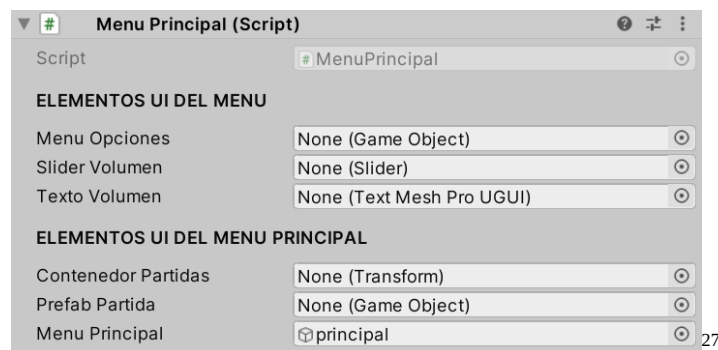
MENUS

Los menús permiten al usuario navegar el sistema para comenzar o terminar una partida, o para ajustar las configuraciones del cliente.

El componente Menú **NO PUEDE SER UTILIZADO POR SÍ SOLO**, es una interfaz para uniformar otros tipos de menús.

CREAR EL MENÚ PRINCIPAL

Es la primera interfaz que puede ver el usuario al abrir la aplicación. Permite navegar al menú de opciones y seleccionar/crear un personaje para jugar una partida.



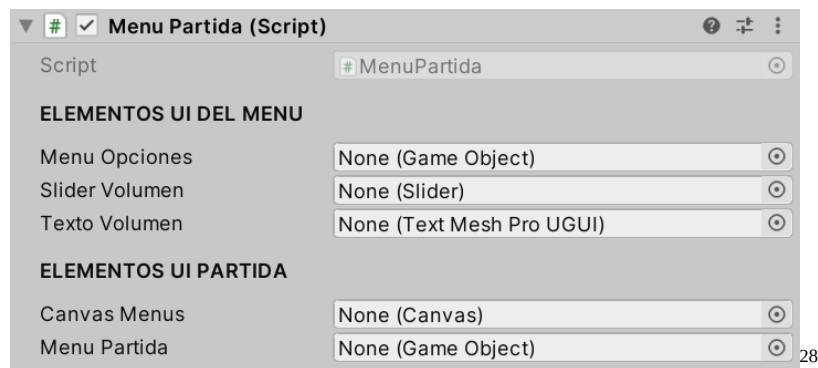
Menú Principal

Componentes expuestos al desarrollador

CREAR EL MENÚ EN PARTIDA

Es la interfaz que permite al jugador navegar al menú de opciones o terminar la sesión de juego, volviendo al menú principal.

²⁷ Ilustración 4.13: Propiedades del Menú Principal(Fuente: Unity Editor 2022.3.5f1)

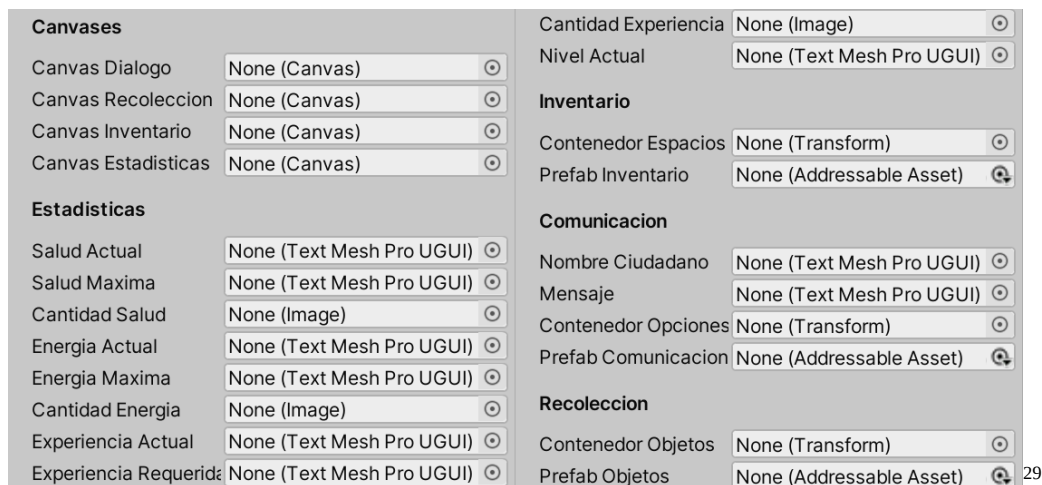


Menú en Partida

Componentes expuestos al desarrollador

INTERFACES DE JUGADOR

Todos los elementos que representen visualmente la información del personaje del jugador en la interfaz de usuario se controlarán desde el mismo script.



Interfaz de Usuario

Referencias a elementos de la interfaz de usuario

²⁸ Ilustración 4.13: Referencias a los elementos del menú de partida del juego. (Fuente: Unity Editor 2022.2.5f1)

²⁹ Ilustración 4.14: Referencias a los elementos de la interfaz de usuario del juego. (Fuente: Unity Editor 2022.3.5f1)

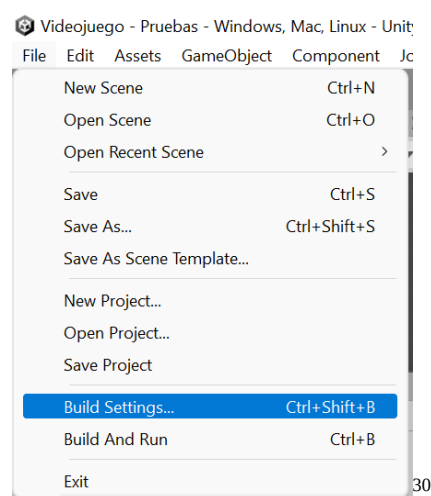
CAPÍTULO 5: COMPILACIÓN A PLATAFORMAS SELECCIONADAS

Actualmente existen muchísimas plataformas en las que experimentar videojuegos, y un sinfín de plataformas que permiten distribuirlos a través de internet. Para este proyecto se ha optado por distribuir la aplicación para la plataforma Windows, y se ha decidido utilizar itch.io como una plataforma gratuita que permite crear una página web para el producto.

COMPILACIÓN A WINDOWS

Por defecto Unity permite la compilación de proyectos para Windows. Cualquier otro tipo de compilador requerido debe ser descargado como un módulo adicional para el editor deseado.

Para compilar un proyecto es necesario seleccionar **File Archivo** en la barra superior, y en el menú desplegado seleccionar **Build Settings Configuración de compilación**.

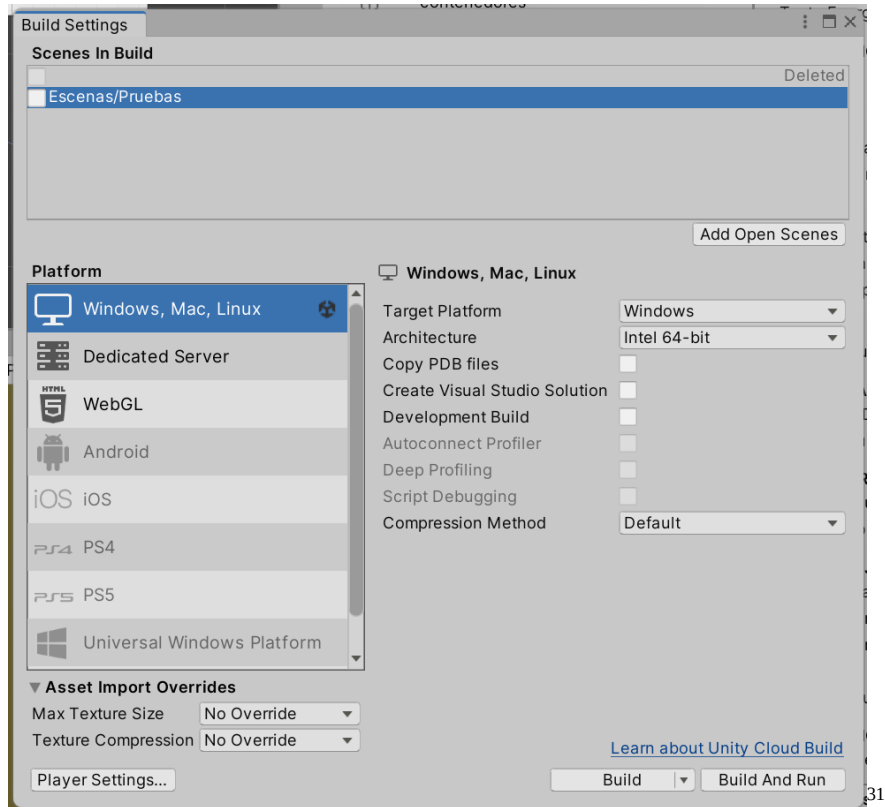


³⁰ Ilustración 5.1: Menú “Archivo” del Editor de Unity. (Fuente: Unity Editor 2022.3.5f1)

Unity Editor

Abriendo el configurador de compilación.

En la ventana desplegada se seleccionan las escenas que queremos utilizar en nuestro programa, y la plataforma objetivo a la que queremos compilar.



Unity Editor

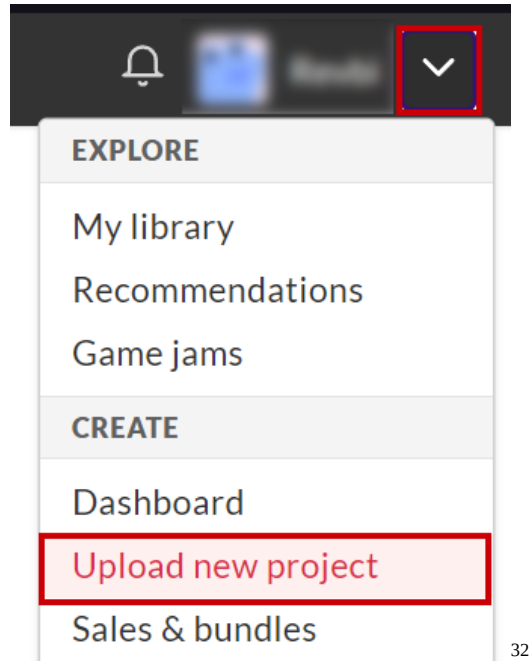
Configurando la compilación de un proyecto.

DISTRIBUCIÓN EN LÍNEA

Para distribuir un proyecto en itch.io es necesario primero crear una cuenta, esto se puede hacer en <https://itch.io/register>.

³¹ Ilustración 5.2: Configuración de compilación. (Fuente: Unity 2022.3.5f1)

Una vez creada la cuenta e iniciada la sesión, desplegar el menú de usuario, haciendo click en la esquina superior derecha de la ventana y se selecciona **Upload new project** *Subir nuevo proyecto*.



Itch.io

Menú de usuario

En la siguiente ventana el desarrollador podrá personalizar toda la información del proyecto, adjuntar imágenes, videos y asignar un valor a su producto.

³² Ilustración 5.3: Menú de usuario registrado (Fuente: itch.io)

Dashboard

Create a new project

Make sure everyone can find your page
Review our [quality guidelines](#) before posting your project

Title

Project URL

Short description or tagline
Shown when we link to your project. Avoid duplicating your project's title

Classification
What are you uploading?

Games — A piece of software you can play

Upload Cover Image

The cover image is used whenever itch.io wants to link to your project from another part of the site. Required (Minimum: 315x250, Recommended: 630x500)

Gameplay video or trailer
Provide a link to YouTube or Vimeo.

eg. <https://www.youtube.com/watch?v=5JEaA47sP>

Screenshots
Screenshots will appear on your game's page. Optional but highly recommended. Upload 3 to 5 for best results.

33

Itch.io

Creando un nuevo proyecto.

³³ Ilustración 5.4: Creando un nuevo proyecto en Itch.io (Fuente: itch.io)

CONCLUSIONES

El sistema propuesto responde a las necesidades planteadas por UwO Studios, permitiendo, a través de Unity, generar niveles ubicando en ellos entidades que **puedan** ayudar o presentar desafíos para el jugador y una estructura de archivos que pueden usar tanto en este como cualquier otro proyecto desarrollado con Unity.

La selección de Unity como motor de desarrollo facilitó mucho la implementación de las mecánicas en el videojuego debido a la extensa red de soporte provisto por Unity Technologies y la comunidad.

Como cualquier otro proceso de desarrollo de tecnologías de información, el desarrollo de videojuegos se beneficia enormemente de buen diseño y planeación.

REFERENCIAS

- Bell, D. (14 de Septiembre de 2004). The UML 2 class diagram. *IBM Developer*. Obtenido de <https://developer.ibm.com/articles/the-class-diagram/>
- Chen, P. P.-S. (Marzo de 1976). The Entity-Relationship Model - Towards a Unified View of Data. (C. M. Jermaine, Ed.) *ACM Transactions on Database Systems*, 9-36. Obtenido de <https://dl.acm.org/doi/abs/10.1145/320434.320440>
- Granberg, A. (s.f.). *A* Pathfinding Project*. Obtenido de <https://arongranberg.com/astar/features>
- itch corp. (2022). *itch.io*. Recuperado el 2022, de itch.io
- Schardon, L. (22 de Febrero de 2022). *GameDev Academy*. Obtenido de <https://gamedevacademy.org/what-is-unity/>
- Unity Technologies. (2022). *Unity*. Obtenido de <https://unity.com/download#:~:text=What%20is%20the%20Unity%20Hub%3F>
- Unity Technologies. (2022). *Unity*. Obtenido de <https://docs.unity3d.com/Manual/BestPracticeUnderstandingPerformanceInUnity6.html>

ANEXOS

A. DOCUMENTO DE DISEÑO DEL JUEGO

VISIÓN GENERAL

Mientras dura la partida el jugador toma el control de un sacerdote - nombrado al comienzo de la partida - en la búsqueda de su hermano perdido en el **Bosque Lóbrego**. A medida que se adentra en el bosque, siguiendo las rutas comerciales que conectan las colonias establecidas por los seguidores de **Ploenut** para extraer materiales valiosos, el sacerdote se verá enfrentado a los horribles y peligrosas criaturas que hacen del bosque su hogar.

OBJETIVO DEL JUEGO

El **propósito inicial y principal** del jugador es encontrar a su hermano perdido en lo profundo de un bosque, pero es posible encontrar objetivos secundarios explorando el mundo.

GÉNERO

Se presenta al jugador con desafíos de combate, donde memorizar los patrones de ataques enemigos - y tener buenos reflejos - podrían significar la diferencia entre progresar o ser derrotado. También se presenta un mundo semi-abierto donde el jugador puede encontrar diversas recompensas como recompensas por explorar el mapa.

Estos aspectos calificarían al juego con los géneros de **acción y aventura**.

AUDIENCIA

El público que se busca alcanzar no diferenciar género, ni edades, sin embargo el despliegue de **violencia de fantasía** podría darle un rating de **Ocho años o más** en la tabla de clasificación por edades de

videojuegos del dispuesta, en parte, por el **Ministerio de Economía, Fomento y Turismo de Chile**, en vez de un rating de **Para todas las edades**.

PLATAFORMAS

El videojuego debe ser compilado y ejecutable para computadoras de escritorio, teniendo el mouse y teclado como esquema de controles principal.

HISTORIA

La historia se centra en el menor de tres hermanos, un joven ordenado sacerdote y retirado en un enclaustro, recibe la noticia de que su hermano mayor, quien se dedicaba a la brujería ha desaparecido. Su otro hermano, quien vivía en los bosques, decide ir a buscarlo, y pese a insistirle a su hermano menor que le acompañara, ya que este llevaba siempre consigo la protección de Dios, este se rehúsa y el leñador parte solo.

Meses después recibe la noticia de que su hermano ha muerto en la búsqueda. Carcomido por la culpa de la muerte de su hermano, el sacerdote decide internarse en los bosques para terminar la misión de su hermano y encontrar a su hermano mayor e intentar traerlo de vuelta al camino de la luz.

JUGABILIDAD

COMBATE

El jugador podrá **equipar** armas desde su inventario para hacer frente a las criaturas del bosque que intentan derrotarlo. Cada ataque del arma drena una cantidad de **energía** del personaje del jugador y crea un área que hace daño a los personajes con los que hace contacto.

Los personajes pueden **esquivar** para evitar el daño de ataques a cambio de **energía**, lo que también incrementa su velocidad de movimiento mientras dura la acción.

Los **enemigos** intentarán atacar al jugador si son incitados por un ataque del jugador, o si pueden mantener contacto visual con él por cierta cantidad de tiempo, momento en el que comenzarán a perseguirlo y atacar si este entra en sus respectivos rangos de ataques.

Una vez derrotados los enemigos podrían crear contenedores con hasta tres objetos de un listado de objetos, ordenados por la probabilidad de obtener determinadas cantidades de algún **objeto**.

INTERACCIONES

Algunos objetos del mundo presentarán al jugador con un mensaje indicando que puede interactuar con este, qué acción se va a llevar a cabo y el nombre del objeto con el que va a interactuar.

Los tipos de interacciones disponibles serán:

Abrir → Contenedor

1. Muestra todo el contenido de un contenedor.
2. Se le indica al jugador que puede recoger todo el contenido con pulsando nuevamente el botón de interacción.
3. Se le indica

Ejemplos: Abrir Baúl, Abrir Mochila, Saquear Enemigo, Cortar Árbol, Minar Hierro, Recoger Basura, Recoger Espada, etc.

4. Hablar → Personaje

- a. El jugador inicia una conversación con un **Personaje**.
- b. Muestra el nombre del personaje y su
- c. Se muestran las **opciones de diálogo** disponibles para esta conversación particular.
- d. Se muestra una opción adicional etiquetada como Adios que termina la interacción.

- e. Si el emisor objetivo tiene misiones disponibles, estas se deben presentar como una variante de opciones de respuesta.

Ejemplos: Hablar Herrero, Hablar Guardia, Leer Libro, etc.

MOVIMIENTO

Todos los personajes están limitados a un movimiento de dos dimensiones. El jugador puede navegar a su personaje a cualquier lugar dentro de los confines del mapa, navegando alrededor de obstáculos. La velocidad de su desplazamiento es incrementada con el nivel de su personaje.

Idealmente los personajes mirarán en la dirección que se están desplazando actualmente. El personaje del jugador debe mirar en dirección a la posición del cursor en la pantalla.

INVENTARIO

El inventario es un contenedor persistente de objetos donde el jugador almacena sus pertenencias - armas, objetos curativos, entre otros. Desde aquí puede seleccionar un objeto para utilizarlo, equipando armas, o utilizando objetos curativos para recuperar salud, por ejemplo. Cada objeto tendrá un peso y la cantidad de peso que puede llevar el personaje del jugador en su inventario dependerá de su nivel...

$$\text{PesoMáximo} = 100 + \text{nivel} \times 1.1f$$

EXPERIENCIA DE USUARIO (UX)

Al iniciar el juego, el usuario se encontrará en el menú principal, donde podrá encontrar tres **espacios de partida**, un botón para ir a las **opciones del cliente**, y otro para **terminar el juego**.

Si no ha creado ninguna partida con anterioridad, entonces deberá crear una nueva partida. Si tiene algún personaje creado, el nombre de este, junto a su nivel, aparecerá en el menú, y el jugador podrá seleccionarlo para continuar su aventura.

Cuando selecciona un personaje se muestra una pantalla de carga y se ubica al jugador en el último mapa y posición donde se encontraba su personaje. Si está comenzando una nueva partida con un nuevo personaje, entonces se cargará al personaje en un mapa y posición predeterminada.

Una vez dentro de la partida el jugador será guiado para aprender las mecánicas del juego, dominarlas dependerá de ella o él.

Las comunidades encontradas estarán construidas de tal forma que personajes importantes sean fácilmente identificables y encontrados.

Dentro de la partida el jugador podrá abrir un menú para guardar su partida actual, ver el menú de opciones o terminar su partida. Dentro del menú de opciones el jugador podrá cambiar el volumen general del juego.

INTEFACES DE USUARIO



MENÚ PRINCIPAL, CON UNA PARTIDA GUARDADA

Muestra el nombre del proyecto, espacios para tres partidas, un botón para salir del juego, y un botón para mostrar las opciones.

Se muestran los tres espacios de partidas, el primero ocupado por la partida de un jugador con un personaje de nivel 17, y los otros dos vacíos.



MENÚ EN PARTIDA

Muestra el nombre del proyecto, un botón para continuar jugando, otro botón para mostrar las opciones, y por último botón para salir del juego.

Al seleccionar continuar, se cierra el menú y la partida continúa normalmente. Al seleccionar salir, se guarda el estado del jugador actualmente y se interrumpe la partida, volviendo al menú principal.



ESTADÍSTICAS DEL JUGADOR

Muestra el nivel del personaje, una barra con el porcentaje de experiencia hasta subir de nivel, las barras de salud y energía, con la cantidad actual y máxima de cada recurso sobre las barras correspondientes, y el ataque equipado actualmente.

La barra de experiencia muestra las estadísticas de un personaje nivel 18, con un 50% de experiencia restante para subir de nivel, 5 puntos de salud de un máximo de 10, un punto de salud de un máximo de 10, equipado con una Espada.



INVENTARIO DEL JUGADOR

Muestra el título de la ventana y todos los objetos que lleva el jugador.

Los objetos se muestran como un listado, donde se agrupan todos los objetos similares en un elemento, mostrando una imagen del objeto, su nombre, y la cantidad de veces que se encuentra en el inventario.



ESTADÍSTICAS DE LOS ENEMIGOS

Muestra un enemigo con su nivel, porcentaje de peligro y cantidad de salud.

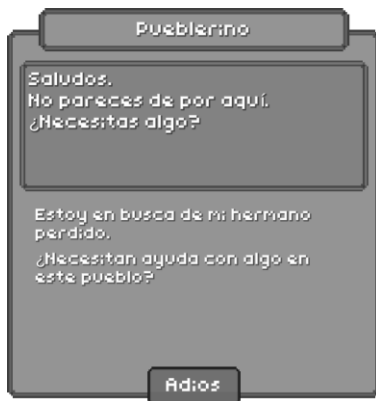


El enemigo lleva viendo al jugador aproximadamente un 75% del tiempo requerido para comenzar a seguirlo, es nivel 5, y tiene el 100% de su salud.

INTERFAZ DE INTERACCIÓN

Muestra un mensaje indicando el verbo y objetivo de la interacción a tomar lugar.

INTERFAZ DE DIÁLOGO



Muestra el nombre del personaje con el que se está dialogando, el mensaje que se le entrega al jugador, las opciones con las que este puede responder, y un botón para terminar la conversación.