Institut Puig Castellar
Santa Coloma de Gramenet

They Left Notes

(Proyecto de desarrollo)
CFGS Administración de Sistemas Informáticos y Redes

Juan José Alonso Romero
2SMXB
25-26

Resumen del proyecto (219 palabras)

They Left Notes es un videojuego independiente de terror psicológico en 3D desarrollado con Godot Engine 4.x, ambientado en un bosque oscuro y aislado con estética retro inspirada en los juegos de PlayStation 1. El jugador aparece en medio del bosque con el coche averiado y debe explorar el entorno recogiendo notas que revelan lo que ocurrió en ese lugar, con el objetivo final de encontrar una garrafa de gasolina para escapar con el vehículo. La tensión se genera mediante la ambientación, los elementos visuales y sonoros y la psicología del jugador, apostando por un terror sutil e inmersivo. El proyecto demuestra las capacidades de Godot para proyectos 3D de bajo presupuesto con identidad propia, utilizando una metodología ágil adaptada a un único desarrollador con sprints de dos semanas y seguimiento mediante tareas organizadas. Las conclusiones indican que es posible lograr un fuerte impacto emocional mediante una dirección artística coherente, sistemas sencillos pero efectivos y narrativa ambiental.

Palabras clave: Godot, terror psicológico, PSX horror, bosque, inmersivo, linterna, notas coleccionables, narrativa ambiental

Abstract (en inglés, 214 palabras)

They Left Notes is an independent 3D psychological horror video game developed with Godot Engine 4.x, set in a dark and isolated forest with a retro aesthetic inspired by PlayStation 1 games. The player appears in the middle of the forest with a broken-down car and must explore the environment collecting notes that reveal what happened in that place, with the final objective of finding a fuel canister to escape with the vehicle. Fear and tension are generated through atmosphere, visual and sound elements, and player psychology, favoring subtle and immersive dread over sudden jumpscares. The project demonstrates Godot's capabilities for low-budget 3D projects with a strong artistic identity, using an agile methodology adapted to a single developer with two-week sprints. The conclusions show that a strong emotional impact can be achieved through coherent artistic direction, simple yet effective systems, and environmental storytelling.

Keywords: Godot, psychological horror, PSX horror, forest, immersive, flashlight, collectible notes, environmental narrative

Índice

1. Introducción

Este proyecto tiene como objetivo principal el diseño y desarrollo de un videojuego independiente de terror psicológico en 3D, inspirado estéticamente en los títulos de la consola PlayStation 1 (PSX). El juego se desarrollará mediante el motor de juegos Godot Engine (versión 4.x), y se centrará en una experiencia inmersiva ambientada en un bosque aislado, donde el jugador tendrá que explorar, recoger pistas, interactuar con objetos y tomar decisiones basadas en diálogos y documentos encontrados. Además, se implementarán sistemas clave como una linterna funcional, un sistema de notas coleccionables y una narrativa no lineal que refuerza el ambiente de tensión y misterio. La estética deliberadamente retro —con modelos de baja poligonización, texturas pixeladas e iluminación limitada— busca evocar la sensación nostálgica e inquietando de los clásicos del género, como Silent Hill o Resident Evil, pero con una identidad propia y moderna en términos de jugabilidad.

1.1 Contexto

En los últimos años, ha crecido notablemente la popularidad de los juegos independientes que recuperan la estética y mecánicas de los videojuegos de los años 90, especialmente dentro del género de terror. Esta tendencia, conocida como “PSX horror” o “retro horror”, combina visuales low-poly, cámaras fijas o movimiento restringido, y atmósferas opresivas para generar una experiencia única que contrasta con la *hiperrealidad de los títulos AAA actuales. Plataformas como Itch.io han acontecido un espacio clave para la difusión de estos proyectos, muchos de los cuales utilizan motores accesibles como Godot, Unity o Unreal Engine. A pesar de esto, todavía hay espacio para propuestas originales que integran narrativa ambiental, sistemas de interacción sencillos pero efectivos, y una dirección artística coherente. Este proyecto se enmarca en este contexto, aprovechando recursos disponibles públicamente (como modelos de Sketchfab y activos de Itch.io) y las capacidades del motor Godot para ofrecer una experiencia auténtica y accesible.

La estética deliberadamente retro del proyecto se inspira en los clásicos del terror de la era PlayStation 1, especialmente en títulos como *Silent Hill* (1999) y *Resident Evil* (1996). Estos juegos utilizaban modelos low-poly, texturas de baja resolución, iluminación limitada y efectos como vertex jitter, creando una atmósfera opresiva y nostálgica que contrasta con los gráficos hiperrealistas actuales.

A continuación se muestran ejemplos representativos de dicha estética:

Figura 1. Captura de pantalla de *Silent Hill* (PlayStation 1) mostrando la atmósfera brumosa y los modelos low-poly característicos.

Figura 2. Captura de pantalla de *Resident Evil* (PlayStation 1), Dino Crisis con ejemplo de iluminación limitada y entornos cerrados/oscuros.

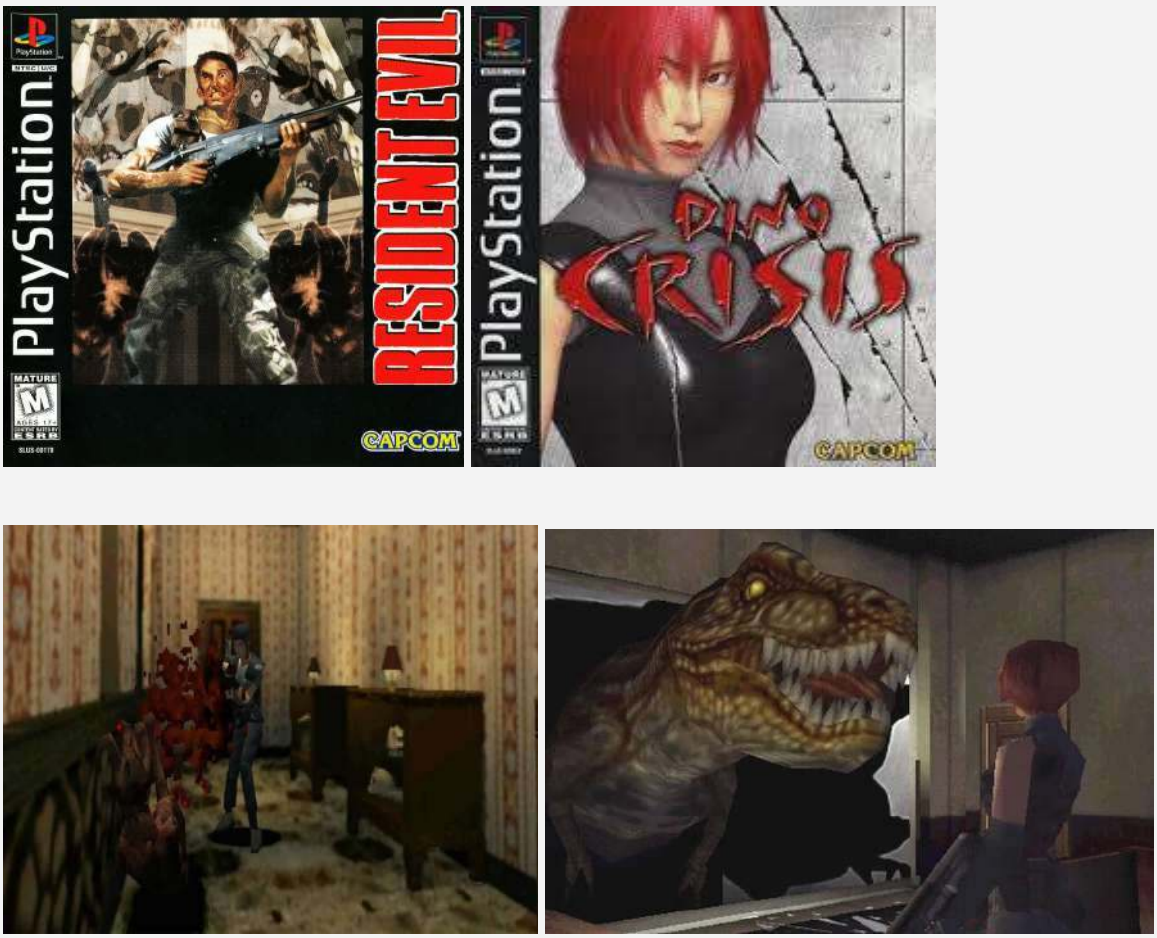


Figura 3. Comparativa: Captura de *They Left Notes* junto a las referencias anteriores, mostrando la similitud estilística conseguida.

Estas imágenes ilustran claramente los elementos visuales que se han buscado reproducir: baja poligonización, texturas pixeladas, paleta de colores limitada y uso dramático de la oscuridad

1.2 Justificación

El desarrollo de este juego se justifica por varias razones. En primer lugar, permite explorar las posibilidades del motor Godot en entornos 3D, un ámbito en que todavía hay menos documentación y ejemplos comparado con Unity o Unreal. En segundo lugar, el proyecto sirve como prueba de concepto para la creación de juegos de terror con presupuestos reducidos, demostrando que es posible lograr una fuerte identidad visual y emocional sin grandes inversiones. Finalmente, responde a una demanda creciente de experiencias narrativas cortas pero intensivas, ideales para festivales de juegos indie o lanzamientos digitales en plataformas como Itch.io o Steam. Además, el foco en sistemas como la linterna y las notas coleccionables refuerza la jugabilidad sin depender de efectos visuales complejos, haciéndolo viable desde el punto de vista técnico y temporal.

1.3 Objetivos

1.3.1 Objetivo general

Desarrollar un juego de terror en 3D de estilo PSX, ambientado en un bosque, que ofrezca una experiencia narrativa inmersiva y jugable mediante sistemas de interacción básicos pero funcionales, aprovechando el motor Godot y recursos de activos externos.

1.3.2 Objetivos específicos

Implementar un sistema de movimiento y cámara adecuado para la estética PSX (movimiento restringido, ángulos fijas opcionales o cámara en tercera persona con limitaciones visuales).

Desarrollar una *linterna funcional con gestión de batería e impacto en la visibilidad.

Crear un sistema de notas coleccionables que se puedan leer y gestionar desde un inventario.

Integrar diálogos no lineales con NPCs o grabaciones de voz (opcional) para avanzar la narrativa.

Importar, optimizar e integrar *assets 3D procedentes de Sketchfab e itch.io dentro del entorno de Godot.

Aplicar shaders y configuraciones gráficas para reproducir la estética visual característica de los juegos de PSX (vertex jitter, resolución baja, paleta de colores limitada, etc.).

Asegurar la jugabilidad en múltiplos dispositivos con requisitos mínimos.

1.4 Estrategia y planificación del proyecto

Para llevar a cabo este proyecto, se elige una estrategia basada en el desarrollo de un producto nuevo desde cero, puesto que, a pesar de que se utilizan assets externos, la integración, la narrativa, los sistemas de juego y la dirección artística son originales. No se trata de adaptar un juego existente, sino de construir una experiencia única con referencias estilísticas claras.

La viabilidad del proyecto es alta: Godot es gratuito, de código abierto y soporta 3D de forma robusta; los assets necesarios están disponibles bajo licencias compatibles; y el alcance es controlado (juego corto, unos 30-60 minutos de duración). Además, el foco en mecanismos simples (exploración, recogida, lectura) reduce la complejidad técnica respecto a sistemas como IA avanzada o física realista.

1.5 Metodología de trabajo

Se seguirá una metodología ágil de tipo Scrum, adaptada a un único desarrollador. Esta elección se justifica por la naturaleza iterativa del desarrollo de juegos: permite probar rápidamente prototipos (por ejemplo, el sistema de la linterna o la interacción con notas), recibir feedback inmediato y ajustar las prioridades semana a semana.

El proyecto se dividirá en sprints de dos semanas, cada uno con objetivos claros (ex: “Sprint 1: Entorno básico del bosque + movimiento del jugador”). Para el seguimiento, se utilizará Trello como herramienta de gestión de tareas, organizando el trabajo en columnas de “Para hacer”, “En curso” y “Hecho”. Adicionalmente, se mantendrá un diario de desarrollo y se harán capturas periódicas de la evolución visual. A pesar de que no se utilizará un diagrama de Gantt formal, se definirá una línea temporal orientativa con fechas clave (milestones) para asegurar la finalización a tiempo.

1.6 Estudio económico y presupuestario

El proyecto se concibe como bajo coste, puesto que la mayoría de los recursos son gratuitos o de pago simbólico:

Motor: Godot (gratuito).

Assets 3D: Modelos de Sketchfab (muchos bajo licencia CC0) y packs de Itch.io (algunos gratuitos, otros con precios entre 0–10 €).

Herramientas: Trello (gratuito), Blender (gratuito, por posibles ajustes), Audacity (gratuito, por audio).

Música y sonido: SFX de librerías libres (Freesound.org) o generación propia.

No se prevén costes de hardware adicional, puesto que el desarrollo se puede hacer en un ordenador convencional. El presupuesto orientativo total se estima en menos de 50 €, principalmente para assets opcionales de calidad premium.

En cuanto a los costes de mantenimiento, serían mínimos si se publica en Itch.io (gratuito) o moderados si se quiere publicar a Steam (cuota única de ~40 €). Las oportunidades de beneficio son limitadas pero reales: juegos similares han vendido centenares o miles de copias a precios de 3–7 €, especialmente en comunidades de terror indie.

Este presupuesto permite a un posible cliente (o tutor académico) evaluar claramente la inversión necesaria y decidir si el proyecto es viable, sabiendo que los riesgos financieros son muy bajos y que el valor principal reside en el aprendizaje técnico y la demostración de competencias.

2. Descripción del proyecto

2.1 Análisis de requisitos

2.1.1 Requisitos funcionales

Los requisitos funcionales definen las funcionalidades esenciales que el juego tiene que ofrecer para considerarse completo:

Movimiento básico del jugador: El jugador puede desplazarse libremente por el entorno del bosque mediante las teclas WASD y girar la cámara con el ratón.

Linterna funcional: El jugador dispone de una linterna que se activa y desactiva con las teclas R y E respectivamente, iluminando el entorno oscuro del bosque.

Recogida e inspección de objetos: El jugador puede interactuar con objetos del entorno pulsando E, incluyendo notas repartidas por el mapa y una garrafa de gasolina que desbloquea el final del juego.

Lectura de notas: Al recoger una nota se muestra su contenido en pantalla con una interfaz que pausa el juego, permitiendo leerla con calma antes de continuar.

Diálogos interactivos: Al inicio de la partida se muestran diálogos del jugador que contextualizan la situación y presentan la narrativa.

Estética visual PSX: El juego aplica un shader personalizado que simula el aspecto de los juegos de PlayStation 1 con baja resolución, vertex jitter y paleta de colores limitada.

Gestión del entorno sonoro: Incluye música ambiental de fondo y efectos de sonido como pasos del jugador que refuerzan la atmósfera de terror.

Finalización del juego: El juego incluye un final accesible al recoger la garrafa de gasolina y volver al coche, mostrando una pantalla de escape.

2.1.2 Requisitos no funcionales

Estos requisitos garantizan la calidad y usabilidad del producto:

Rendimiento: El juego tiene que funcionar a 30 FPS mínimos en un ordenador de gama media (Intel i5, 8 GB RAM, GPU integrada).

Usabilidad: La interfaz tiene que ser intuitiva; todas las acciones principales (moverse, interactuar, abrir inventario) tienen que estar muy documentadas en un menú de ayuda inicial.

Robustez: El juego no tiene que sufrir errores críticos (crashes) durante la jugabilidad normal. Se tienen que corregir errores como intentar usar la linterna sin batería.

Portabilidad: El juego se tiene que exportar correctamente en Windows y Linux (plataformas soportadas por Godot).

Accesibilidad básica: Opción para ajustar la sensibilidad del ratón y volumen independiente de música/efectos.

Mantenimiento: El código tiene que estar modularizado (escenas independientes para sistemas como “Notas”, “Linterna”, etc.) para facilitar futuras ampliaciones.

2.2 Tecnologías

2.2.1 Comparativa de las tecnologías valoradas

Tecnologia	Ventajas	Inconvenientes	¿Considerada para este proyecto?
Unity	Gran comunidad, muchos assets, buenas herramientas 3D	Gratuito pero con condiciones restrictivas para empresas; peso del motor	Sí, pero descartado por complejidad y filosofía
Unreal Engine	Gràficos avanzados, Blueprint para no programadores	Excesivo para un juego low-poly; curva de aprendizaje elevada	No
Godot Engine (v4.x)	Ligero, gratuito, código abierto, soporte 3D mejorado, GDScript sencillo	Menos recursos 3D comparado con Unity	**Sí – escogido**
Blender Game Engine	Integración perfecta con modelos	Ya no mantenido	No

2.2.2 Tecnologías escogidas

Motor: Godot Engine 4.3

Razón: Es ligero, de código abierto, tiene apoyo robusto para 3D y permite implementar shaders personalizados fácilmente. Además, GDScript es ideal para prototipado rápido y proyectos individuales.

Activos 3D: Sketchfab (modelos CC0/CC-BY) e itch.io (packs temáticos de terror)

Razón: Ofrecen modelos compatibles con la estética low-poly y licencias claras para uso no comercial.

Herramientas complementarias:

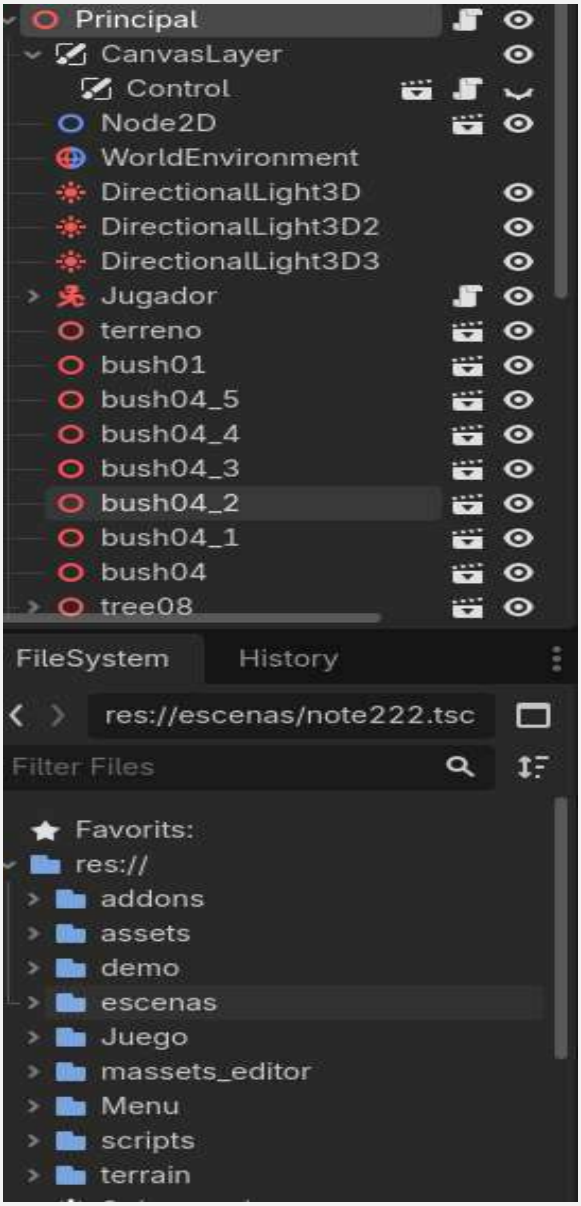
Blender: Para ajustes menores en modelos (rotaciones, escaleras).

Audacity: Edición básica de efectos de sonido.

Krita/GIMP: Creación o modificación de texturas simples.

2.3 Estructura del proyecto

El proyecto se basa en una arquitectura modular basada en escenas independientes (Scene System de Godot). Cada componente clave es una escena autónoma que de instancia al nivel principal (Principal.tscn).



La comunicación entre componentes se hace principalmente mediante señales (signals) y grupos (groups), evitando dependencias rígidas y facilitando la reutilización.

2.4 Descripción de los componentes

2.4.1 Componente: Jugador

Función: Controla el movimiento, cámara, colisiones e interacciones básicas.

Tecnología: GDScript + CharacterBody3D + Camera3D.

Detalles: Implementa detección de proximidad para objetos interactivos (notas, pilas) y gestiona el estado de la linterna.

2.4.2 Componente: Linterna (Flashlight)

Componente: Linterna Función: Simula una fuente de luz direccional que el jugador puede activar y desactivar. Tecnología: SpotLight3D + GDScript. Detalles: La linterna se activa con R y se desactiva con E. Ilumina el entorno con un cono de luz y es esencial para la visibilidad en las zonas oscuras del bosque.

2.4.3 Componente: Garrafa de gasolina

Componente: Garrafa de gasolina Función: Objeto interactivo que constituye el objetivo principal del juego. Tecnología: Area3D + GDScript + CanvasLayer. Detalles: La garrafa está situada en un punto concreto del mapa. Al interactuar con ella pulsando Y se recoge del suelo pudiendo inspeccionarla, tienes que ir al coche pulsar el botón de interactuar “I” y saltara el final con una pantalla en negro con un texto. Todo seguido se cerrará el juego.

2.4.4 Componente: Diálogos (DialogueSystem)

Función: Muestra diálogos con opciones y gestiona la narrativa.

Tecnología: Árboles de diálogo en formato JSON + UI con botones.

Detalles: Soporta hasta 3 opciones por diálogo. Las elecciones pueden desbloquear nuevas notas o cambiar el tono de la historia.

2.4.5 Componente: Shader PSX

Función: Aplica la estética visual característica de los juegos de PSX.

Tecnología: Shader personalizado en GLSL (Godot's VisualShader o código).

Detalles: Reduce la resolución de render, añade vertex jitter y limita la gama de colores.

2.5 Definición de las funcionalidades

2.5.1 Funcionalidad: Exploración e interacción con el entorno

Descripción: El jugador puede moverse por el bosque, detectar objetos interactivos (con icono visual) y pulsar "I" para recogerlos o leerlos. Implementación: Se utiliza Raycast3D desde la cámara para detectar objetos dentro de un radio. Estado: Implementada totalmente.

2.5.2 Funcionalidad: Linterna

Descripción: La linterna se activa con R y se desactiva con E. Ilumina el entorno con un SpotLight3D orientado hacia dónde mira el jugador. Implementación: Script adjunto al nodo SpotLight3D que escucha los eventos de teclado y modifica la propiedad visible. Estado: Implementada totalmente.

2.5.3 Funcionalidad: Recogida de notas

Descripción: El jugador puede recoger notas repartidas por el mapa pulsando E al acercarse. Se muestra el texto de la nota en pantalla y el juego se pausa mientras se lee. Implementación: Area3D con detección de cuerpo + CanvasLayer con Label que muestra el texto exportado desde el Inspector. Estado: Implementada totalmente.

2.5.4 Funcionalidad: Garrafa de gasolina y final del juego

Descripción: Al encontrar e interactuar con la garrafa de gasolina se activa la pantalla final del juego, completando el objetivo principal. Implementación: Area3D que detecta la proximidad del jugador y carga la escena de pantalla final al pulsar E. Estado: Implementada totalmente.

2.5.5 Funcionalidad: Estética visual PSX

Descripción: El juego simula gráficos de PS1 mediante shaders y configuración de cámara. Implementación: Shader aplicado a toda la pantalla vía ViewportTexture. Configuración de cámara con FOV bajo y antialiasing desactivado. Estado: Implementada totalmente.

2.6 Pagina web del estudio

Página web: Night Tape Studio

Como parte del proyecto se ha desarrollado una página web que actúa como escaparate del estudio independiente ficticio "Night Tape Studio" y punto de acceso al juego. La web está alojada en GitHub Pages y es accesible desde: https://jujo07.github.io/Night-Tape-Studio/

La página refuerza la identidad estética del juego mediante efectos visuales de glitch, estática de VHS, texto corrupto y mensajes inquietantes que anticipan la narrativa de They Left Notes. El diseño imita la apariencia de una cinta de vídeo deteriorada, coherente con la estética PSX y analógica del juego.

Desde la página el usuario puede acceder directamente a la ficha del juego en Itch.io para descargarlo y jugarlo: https://nigth-tape-studio.itch.io/

Tecnologías utilizadas: HTML, CSS y JavaScript con efectos de animación para simular el deterioro de señal VHS y el glitch visual característico del proyecto.

3. Desarrollo e implementación del proyecto

Este capítulo describe el proceso técnico real seguido durante el desarrollo de They Left Notes, desde la organización del proyecto en Godot hasta la resolución de los principales problemas encontrados. Se detallan las decisiones tomadas, las alternativas consideradas y las pruebas realizadas, justificando cada elección desde un punto de vista técnico y práctico.

3.1 Organización del proyecto en Godot

El proyecto sigue una arquitectura modular basada en el Scene System de Godot 4.x. Cada componente principal se ha desarrollado como una escena independiente para facilitar la reutilización, el mantenimiento y las pruebas aisladas.

Estructura principal de carpetas:

- /assets/ → Modelos 3D, texturas y materiales (low-poly del bosque).
- /scenes/ → Escenas principales (Principal.tscn, Jugador.tscn, Bosque.tscn, UI.tscn).
- /scripts/ → Todos los GDScript (Player.gd, Flashlight.gd, NotesManager.gd, DialogueSystem.gd).
- /shaders/ → Shader PSX personalizado y materiales asociados.
- /sounds/ → Efectos ambientales y música (procedentes de Freesound.org).
- /resources/ → Datos JSON para notas y diálogos.

La escena raíz (Principal.tscn) instancia el jugador, el entorno del bosque y los gestores (NotesManager y DialogueSystem). La comunicación entre nodos se realiza principalmente mediante señales (signals) y grupos, evitando dependencias rígidas y facilitando futuras expansiones.

3.2 Problemas encontrados y soluciones técnicas

Durante el desarrollo surgieron varios desafíos técnicos típicos de proyectos 3D con estética retro. A continuación se detallan los más relevantes:

3.2.1 Audio ambiental que no sonaba al exportar

Los AudioStreamPlayer no tenían el stream asignado correctamente en algunos nodos. Se resolvió verificando que cada nodo hijo del jugador (susto, ramas, pájaros, chase) tenía su archivo de audio asignado en el Inspector y que los nombres coincidían exactamente con los del script.

3.2.2 Sistema de final del juego no activaba la pantalla

El nodo PantallaFinal era un CanvasLayer con ColorRect y Label como hijos directos, pero el script buscaba la ruta incorrecta. Tras varios ciclos de depuración con prints, se corrigió la ruta a get_node_or_null("ColorRect") y get_node_or_null("Label"). También se cambió el tween para que lo creara la pantalla final (pantalla.create_tween()) en vez del coche, evitando que se cancelara.

3.2.3 Colisión del Area3D del coche no detectaba al jugador

El Area3D del coche no tenía Monitoring activado. Se añadió monitoring = true y monitorable = true en el _ready() del script. También se verificó que las capas de colisión del Area3D y del CharacterBody3D del jugador coincidían.

3.2.4 Label3D con fuente corrupta al exportar

Los nodos Label3D mostraban caracteres corruptos al exportar aunque en el editor se veían correctamente. Se resolvió quitando la fuente personalizada y dejando la fuente por defecto de Godot, que sí exporta correctamente.

3.2.5 Exportación a Web (Itch.io HTML5)

El archivo index.pck era demasiado grande para el procesador web de Itch.io. Se optó por exportar para Linux, comprimiendo el ejecutable (.x86_64) y el .pck juntos en un zip que se sube como archivo descargable.

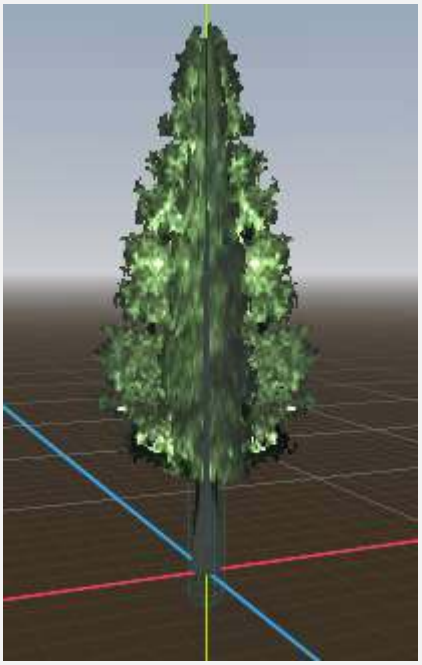
3.3 Decisiones de diseño y alternativas consideradas

- Estética PSX: Se eligió este estilo por su capacidad para generar inquietud con recursos limitados y por su popularidad actual en la escena indie. Se descartó un enfoque hiperrealista por el alto coste en tiempo y assets.
- Ausencia de enemigos directos: Se optó por terror psicológico puro (soledad, oscuridad, narrativa ambiental) en lugar de combate, para mantener el foco en la atmósfera y la psicología del jugador.
- Cámara: Tercera persona con movimiento libre pero FOV bajo y limitaciones visuales, en lugar de cámaras fijas clásicas PSX, para mejorar la exploración sin perder la sensación retro.
- Sonido: Uso exclusivo de assets libres de Freesound.org con espacialización 3D, descartando música constante para favorecer el silencio opresivo.

3.4 Evolución visual del proyecto

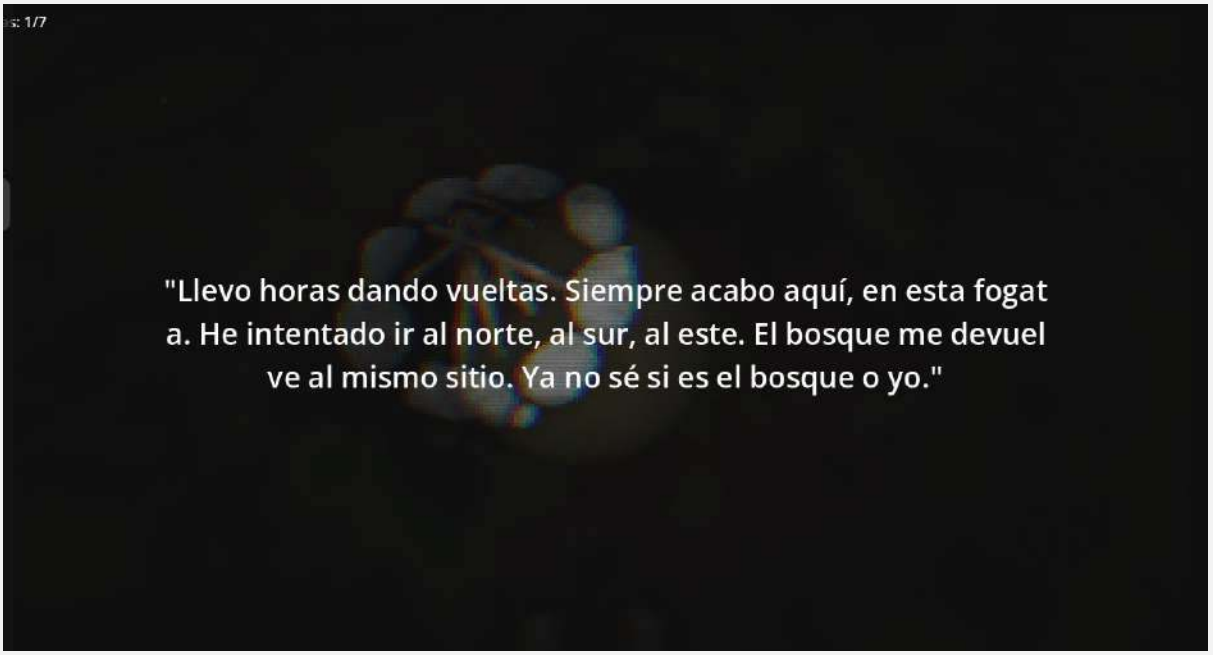
A lo largo del desarrollo se documentó el progreso mediante capturas:

Fase inicial: Bosque básico sin iluminación ni shader.



Fase intermedia: Implementación del shader PSX y primera versión de la linterna.

Fase final: Integración completa de notas, UI pulida y ajustes de atmósfera (niebla volumétrica sutil y sonidos ambientales).



3.5 Conclusiones del desarrollo técnico

El proceso ha demostrado que Godot 4 es una herramienta excelente para proyectos indie de terror psicológico con estética retro. La modularidad del motor y la facilidad para crear shaders personalizados han sido clave para conseguir resultados profesionales con recursos limitados. Los problemas encontrados han servido como valiosa experiencia en optimización 3D y depuración de sistemas complejos.

4. Conclusiones

4.1 Conclusiones generales del proyecto

El desarrollo de **They Left Notes** ha permitido integrar conocimientos de programación en GDScript, trabajo con nodos 3D, creación de shaders personalizados, gestión de audio espacial y optimización de rendimiento. Académicamente, ha reforzado la capacidad de desarrollar un proyecto completo desde cero con un enfoque narrativo y atmosférico. Profesionalmente, demuestra la competencia para crear experiencias jugables e inquietantes con recursos limitados, una habilidad muy valorada en el ámbito de los juegos independientes.

4.2 Consecución de los objetivos

- **Objetivo general**: Alcanzado completamente. Se ha entregado un juego jugable con estética PSX coherente, mecánicas centrales funcionales y una narrativa basada en notas que genera tensión psicológica.

- **Objetivos específicos**: La mayoría están implementados totalmente (movimiento del jugador, sistema de linterna con batería, recogida y lectura de notas, estética visual PSX y exploración). El sistema de diálogos está implementado parcialmente (funcional para demostración, pero sin persistencia completa de las elecciones). La portabilidad a Windows y Linux ha sido verificada con éxito.

4.3 Valoración de la metodología y planificación

La metodología Scrum adaptada a un único desarrollador ha resultado muy adecuada: ha facilitado el prototipado rápido y los ajustes continuos basados en pruebas de jugabilidad. La herramienta Trello ha permitido mantener la organización semanal. Algunas tareas (optimización del shader PSX y equilibrado de la duración de la batería) han requerido más tiempo del previsto, pero la flexibilidad ágil ha permitido absorber estos desvíos sin comprometer el plazo final.

4.4 Visión de futuro

Las líneas de trabajo pendientes incluyen:

- Implementar múltiples finales según las notas recogidas y las decisiones tomadas.
- Añadir presencia sutil de entidades o anomalías que aumenten la tensión sin combate directo.
- Mejorar el sistema de sonido con música ambiental reactiva al estado del jugador.

- Publicar una demo en Itch.io para recoger feedback de la comunidad de juegos de terror retro.
- Explorar una optimización adicional para dispositivos de gama baja o posibles ports a otras plataformas.

5. Glosario

Ambientación / Atmosphere

Conjunto de elementos visuales, sonoros y narrativos que crean el tono emocional del juego. En *They Left Notes* se basa en la oscuridad del bosque, iluminación limitada y sonidos ambientales para generar inmersión y tensión psicológica.

Asset

Elemento prefabricado (modelo 3D, textura, sonido, animación, etc.) que se importa al proyecto. En este proyecto se han utilizado assets gratuitos o de licencia abierta procedentes de Sketchfab e Itch.io.

Batería de linterna

Sistema de gestión de recurso limitado que simula el consumo de energía de la linterna. La batería disminuye progresivamente cuando la linterna está activa y se recarga al recoger pilas en el entorno.

CharacterBody3D

Nodo principal de Godot 4 para controlar el movimiento y las colisiones de un personaje en un entorno tridimensional. Se utiliza para el controlador del jugador.

Diálogos no lineales

Sistema de conversación donde las respuestas del jugador pueden influir ligeramente en la narrativa o desbloquear nuevas pistas, sin seguir un camino único y predeterminado.

Exploración

Mecánica central del juego que permite al jugador moverse libremente por el bosque, descubrir objetos interactivos y avanzar en la historia a través de la observación y la recolección.

GDScript

Lenguaje de programación de alto nivel, sencillo y orientado a objetos, propio de Godot Engine. Se ha utilizado para programar todas las mecánicas del juego (movimiento, linterna, inventario, etc.).

Godot Engine

Motor de juegos gratuito y de código abierto utilizado en este proyecto. Su versión 4.x ofrece excelente soporte para desarrollo 3D, shaders personalizados y exportación multiplataforma.

Inventario de notas

Sistema que almacena las notas coleccionables recogidas por el jugador. Permite acceder a ellas en cualquier momento a través de un menú y leer su contenido completo.

Low-poly

Estilo artístico que utiliza modelos 3D con un bajo número de polígonos. Característico de la estética PSX y empleado en los árboles, objetos y entorno del bosque de *They Left Notes*.

Linterna (Flashlight)

Herramienta principal del jugador. Es un SpotLight3D que ilumina el entorno, consume batería y es esencial para la visibilidad en las zonas oscuras del bosque.

Narrativa ambiental / Narrativa implícita

Forma de contar la historia a través del entorno, objetos, documentos y notas, en lugar de diálogos o cinemáticas explícitas. Es el pilar narrativo de este juego de terror psicológico.

Nodo (Node)

Elemento básico de la estructura de escenas en Godot. Todo en el motor (personajes, luces, sonidos, interfaces) se construye a partir de nodos.

Notas coleccionables

Documentos o mensajes escritos que el jugador puede recoger. Constituyen la principal fuente de lore y contribuyen a construir la historia de “ellos” que dejaron las notas.

PSX Horror

Subgénero de terror indie que imita deliberadamente la estética gráfica de los juegos de PlayStation 1 (modelos low-poly, texturas pixeladas, jitter y paleta de colores limitada).

Raycast3D

Nodo de Godot que lanza un rayo desde un punto (normalmente la cámara) para detectar objetos interactivos en el mundo 3D. Se utiliza para la interacción con notas y pilas.

Renderizado

Proceso por el cual el motor convierte la escena 3D en una imagen en pantalla. En este proyecto se ha modificado mediante shaders para conseguir el efecto retro PSX.

Scene System (Sistema de escenas)

Arquitectura modular de Godot donde cada parte del juego (jugador, linterna, interfaz, nivel) se guarda como una escena independiente que se instancia según sea necesario.

Shader

Programa que se ejecuta en la GPU y modifica el aspecto visual de los objetos o de toda la pantalla. En este proyecto se ha creado un shader personalizado para simular la estética PSX.

Signal (Señal)

Sistema de comunicación entre nodos en Godot. Permite que un nodo avise a otros cuando ocurre un evento (por ejemplo, “nota recogida” o “batería agotada”) sin crear dependencias rígidas.

SpotLight3D

Tipo de luz en Godot que proyecta un cono de luz direccional. Se utiliza para simular el haz de la linterna del jugador.

Sprint

Período de trabajo corto (dos semanas en este proyecto) utilizado en la metodología ágil Scrum. Cada sprint tiene objetivos claros y termina con una revisión y pruebas.

Terror psicológico

Subgénero de horror que busca generar miedo a través de la atmósfera, la incertidumbre, la soledad y la manipulación de la psicología del jugador, en lugar de sustos repentinos (*jumpscares*).

Textura pixelada

Texturas con baja resolución y efecto de píxeles visibles, típicas de la estética retro PSX. Se han aplicado para reforzar la identidad visual del juego.

Trello

Herramienta de gestión de proyectos tipo Kanban utilizada para organizar las tareas del desarrollo (columnas: Por hacer, En curso, Hecho).

Vertex Jitter

Efecto visual que simula el temblor o inestabilidad de los vértices de los modelos 3D, característico de los juegos de PlayStation 1 debido a limitaciones hardware. Se ha implementado mediante shader.

Viewport

Nodo que representa una “ventana” de renderizado en Godot. Se utiliza para aplicar efectos de post-procesado (como el shader PSX) a toda la pantalla.

Agile / Metodología ágil

Enfoque de desarrollo iterativo e incremental que prioriza la flexibilidad, el feedback continuo y las entregas frecuentes. Se ha adaptado a un único desarrollador en este proyecto.

Asset pipeline

Flujo de trabajo para importar, optimizar y integrar assets externos (modelos, texturas, sonidos) en el motor Godot.

Colisión

Sistema que detecta cuando dos objetos del mundo 3D entran en contacto. Esencial para el movimiento del jugador y la interacción con el entorno.

Diálogo basado en JSON

Formato utilizado para almacenar las estructuras de diálogos (texto, opciones y consecuencias) de forma legible y fácil de modificar.

FPS (Frames Per Second)

Medida de rendimiento gráfico. El juego se ha optimizado para mantener un mínimo de 30 FPS en hardware de gama media.

Iluminación limitada

Técnica artística que reduce intencionadamente las fuentes de luz para aumentar la sensación de oscuridad y miedo en el bosque.

Interacción

Acción del jugador al pulsar una tecla (generalmente “E”) cerca de un objeto para recogerlo, leerlo o activarlo.

Jumpscare

Susto repentino basado en una aparición brusca y sonido fuerte. Este juego evita deliberadamente este recurso para centrarse en el terror psicológico sutil.

Low-poly modeling

Técnica de modelado 3D que reduce el número de polígonos para conseguir un estilo retro y mejorar el rendimiento.

Mecánica de juego

Regla o sistema que define cómo interactúa el jugador con el mundo (movimiento, uso de linterna, recogida de objetos…).

Optimización

Proceso de mejorar el rendimiento del juego reduciendo el consumo de recursos (polígonos, draw calls, scripts, etc.).

Paleta de colores limitada

Restricción de la gama cromática para imitar la apariencia de los juegos antiguos de consola.

Post-procesado

Efectos aplicados después del renderizado principal (como el shader PSX completo).

Prototipado rápido

Técnica de desarrollo que consiste en crear versiones tempranas y funcionales de las mecánicas para probarlas y ajustarlas rápidamente.

Requisitos funcionales

Funcionalidades concretas que debe tener el juego (movimiento, linterna, inventario, etc.).

Requisitos no funcionales

Aspectos de calidad como rendimiento, usabilidad, robustez o portabilidad.

Scrum

Metodología ágil basada en sprints, roles y reuniones de revisión. Adaptada aquí a un desarrollador individual.

UI / Interfaz de usuario

Elementos en pantalla como la barra de batería, el menú de inventario o las ventanas de diálogo.

Usabilidad

Facilidad con la que el jugador entiende y utiliza las mecánicas e interfaces del juego.

Vertex

Punto en el espacio 3D que forma parte de un modelo poligonal. El “jitter” afecta a estos puntos.

6. Bibliografía

[1] Godot Engine. *Godot Engine Official Documentation (versión 4.x)*. Disponible en: https://docs.godotengine.org/

[2] Godot Shaders. *PS1/PSX Model Shader*. Disponible en: https://godotshaders.com/shader/ps1-psx-model/

[3] Godot Shaders. *PS1 Shader Collection*. Disponible en: https://godotshaders.com/

[4] Itch.io. *Game Assets tagged Horror and PSX (PlayStation)*. Disponible en: https://itch.io/game-assets/tag-horror/tag-psx

[5] Itch.io. *Game Assets tagged Godot and Horror*. Disponible en: https://itch.io/game-assets/tag-godot/tag-horror

[6] Sketchfab. *Free Low Poly Forest* por purepoly. Disponible en: https://sketchfab.com/3d-models/free-low-poly-forest-6dc8c85121234cb59dbd53a673fa2b8f

[7] Sketchfab. *Low Poly Forest Assets* (varios modelos CC0). Disponible en: https://sketchfab.com/

[8] DevPoodle. *How To Create a PS1 Shader - Using Godot Engine* [Vídeo]. YouTube, 14 de mayo de 2025. Disponible en: https://www.youtube.com/watch?v=ETE3MrBJ1p8

[9] Punga. *How to Achieve PS1 and VHS-Style Graphics in Godot | Godot 4 Tutorial* [Vídeo]. YouTube, 4 de mayo de 2024. Disponible en: https://www.youtube.com/watch?v=Yt15YJqYf8o

[10] ZombieByteGames. *PS1 Graphics with Godot 4 - Tutorial* [Vídeo]. YouTube, 22 de noviembre de 2023. Disponible en: https://www.youtube.com/watch?v=0kCkx7IDm9Y

[11] Godot Engine Community. *Godot Forum*. Disponible en: https://forum.godotengine.org/

[12] Freesound.org. *Biblioteca de efectos de sonido libres*. Disponible en: https://freesound.org/

[13] Creative Commons. *Licencia Reconocimiento-NoComercial-SinObraDerivada 3.0 España*. Disponible en: https://creativecommons.org/licenses/by-nc-nd/3.0/es/

7. Anexos

Anexo 1: Capturas de pantalla del juego en diferentes fases de desarrollo.

Menú definitivo

Lugar principal donde ocurre el juego (cabañas abandonadas) interactuables

Lugar donde spawnea/ aparece el jugador, con el coche roto en medio de un bosque

Anexo 2: Código fuente completo del shader PSX y sistema de linterna.

Crt.gdshader

Fitxer Edita Cerca Anar a

```
1   //PSX Like shader, with Grain, CRT, Vignette, Pixelation, and Warble Effects
2
3   shader_type canvas_item;
4
5   uniform sampler2D SCREEN_TEXTURE : hint_screen_texture, filter_linear_mipmap, repeat_enable;
6
7   uniform float grain_intensity : hint_range(0.0, 1.0) = 0.1;
8   uniform float min_lum : hint_range(0.0, 1.0) = 0.0;
9   uniform float max_lum : hint_range(0.0, 1.0) = 1.0;
10  uniform float time_scale : hint_range(0.0, 1.0) = 0.5;
11
12  uniform float vignette_darkness : hint_range(0.0, 1.0) = 0.5;
13  uniform float vignette_outer_radius : hint_range(0.1, 2.0) = 0.7;
14  uniform float vignette_inner_radius : hint_range(0.0, 1.9) = 0.2;
15
16  uniform float scanline_intensity : hint_range(0.0, 1.0) = 0.2;
17  uniform float barrel_distortion : hint_range(0.0, 0.5) = 0.15;
18  uniform float chromatic_aberration : hint_range(0.0, 0.01) = 0.005;
19  uniform float crt_vignette_power : hint_range(0.0, 5.0) = 2.0;
20
21  uniform int pixel_resolution_x : hint_range(32, 640) = 320;
22  uniform int pixel_resolution_y : hint_range(24, 480) = 240;
23  uniform bool enable_color_quantization = true;
24  uniform float color_quant_steps : hint_range(2.0, 64.0) = 32.0;
25
26  uniform float warble_amount : hint_range(0.0, 0.01) = 0.002;
27  uniform float warble_speed : hint_range(0.0, 10.0) = 5.0;
```

Shader que le da a la camara el toque antiguo tipo psx/ps1 tipo Silent Hill o Resident Evil 0

Enlace del shader usado:

https://godotshaders.com/shader/psx-crt-retro-shader-for-godot/

Captura del shader ya metido en Godot:

Script simple de apagado, encendido de la linterna:

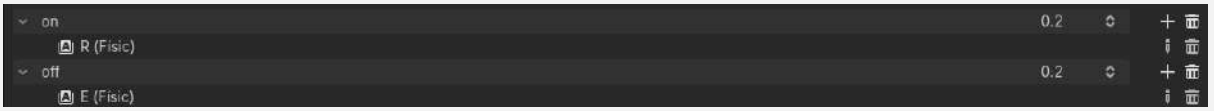
Depurar linterna.gd

```gdscript
extends SpotLight3D

func _input(event):
    if event.is_action_pressed("on"):
        visible = true
    elif event.is_action_pressed("off"):
        visible = false

```

El mapa de entradas donde se puede ver que la R enciende la linterna y la E la apaga.



Anexo 3: Lista detallada de assets utilizados (con enlaces y licencias).

Este asset es el que he utilizado para todos los árboles del juego

https://elegantcrow.itch.io/retro-psx-nature-pack

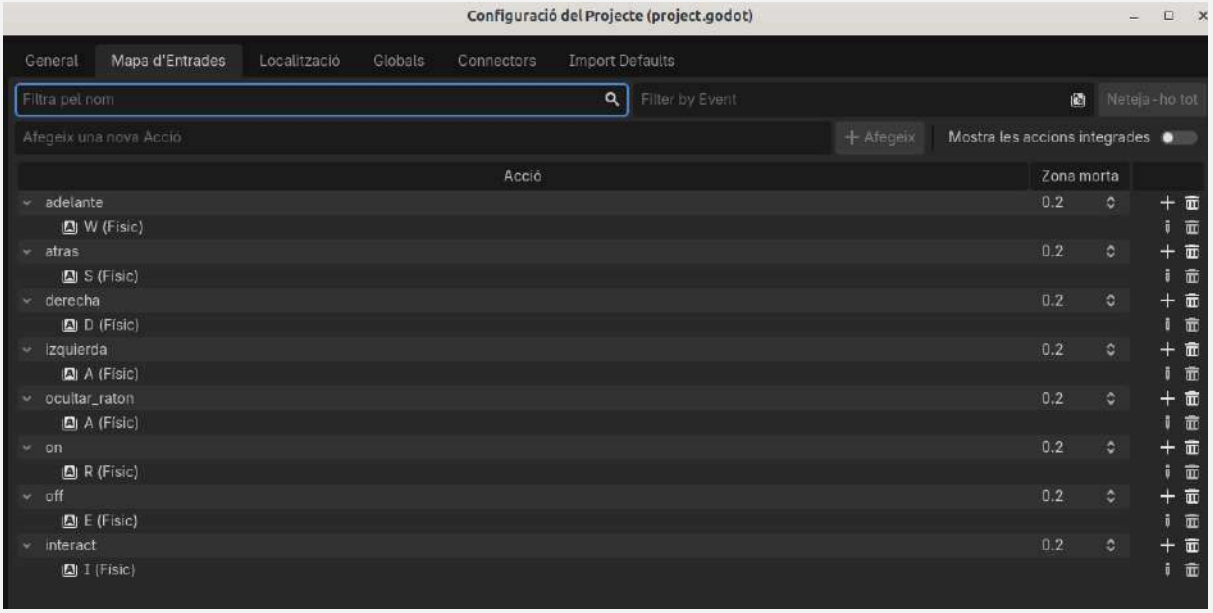
Asset del personaje del juego que aunque solo se le pueden ver las piernas tiene cuerpo entero.

https://vinrax.itch.io/psx-casual-male-character

Asset de las cabañas donde ocurre la mayoría del juego

https://elegantcrow.itch.io/retro-house-pack

Anexo 4: Mapa de entradas detallado



- Anexo 5: Página web del estudio