



UNIVERSIDADE DA CORUÑA

Contornos Inmersivos, Interactivos y de entretenimiento

# DESARROLLO TÉCNICO

Villaverde

José Manuel Amestoy López  
Xoel González Pereira  
Jordi Núñez Arias  
Pablo Fernández Pérez  
Eva María Arce Ale

# Contents

<b>1 Desarrollo Técnico.</b>	<b>2</b>
1.1 Descripción. . . . .	2
1.2 Personajes. . . . .	3
1.3 Herramientas. . . . .	6
1.4 Objetos. . . . .	7
1.5 Diseño. . . . .	9
<b>2 Escenas.</b>	<b>10</b>
2.1 Escena 1: Menú. . . . .	12
2.2 Escena 2: Cinemática. . . . .	14
2.3 Escena 3: Verano . . . . .	15
2.4 Escena 4: Otoño . . . . .	19
2.5 Escena 5: Invierno . . . . .	19
2.6 Escena 6: Créditos. . . . .	21
2.7 Patrones de diseño: . . . . .	21
<b>3 Aspectos destacables.</b>	<b>22</b>
<b>4 Manual de usuario.</b>	<b>25</b>
4.1 Menú principal. . . . .	25
4.2 Objetivos. . . . .	26
4.3 Juego pausado. . . . .	27
4.4 Interacción con NPCs. . . . .	27
4.5 Sistema de tala. . . . .	28
4.6 Sistema de cultivo. . . . .	28
4.7 Sistema de compra-venta. . . . .	29
4.8 Otras interacciones. . . . .	31
4.8.1 Interacción con carteles. . . . .	31
4.8.2 Interacción con la puerta de la mazmorra. . . . .	31
4.9 Puzle. . . . .	31
4.10 Librerías que hay que instalar . . . . .	32
<b>5 Reporte de bugs.</b>	<b>33</b>

# **1 Desarrollo Técnico.**

## **1.1 Descripción.**

- Breve descripción del videojuego:

La ambientación de este juego se sitúa en un entorno rural destrozado, donde los jugadores tienen la oportunidad de sumergirse en la vida en una granja y su reconstrucción. El juego ofrece la posibilidad de realizar actividades agrícolas, que incluyen el cultivo de cultivos, la gestión de animales y la compra-venta de productos. Los jugadores pueden disfrutar de la gestión de su granja, experimentando con diferentes estrategias para alcanzar unos objetivos prefijados. El juego tiene un estilo pixel art muy colorido.

- Objetivos de la memoria:

Documentar el proceso de desarrollo del videojuego de granja, desde su concepción hasta su lanzamiento. Detallar las mecánicas principales y el diseño del videojuego, incluyendo la historia, personajes, entornos y elementos de jugabilidad. Explorar los patrones de diseño utilizados en el desarrollo del videojuego y justificar su elección para diferentes sistemas del juego. Identificar y describir los bugs encontrados durante el desarrollo del videojuego, junto con las soluciones implementadas o propuestas. Proporcionar referencias a recursos utilizados durante el desarrollo y reconocer el trabajo de terceros, como assets o plugins utilizados en el videojuego.

## 1.2 Personajes.

- **Wuan Villaverde (protagonista y jugador):**

Wuan es el personaje principal, presente en todos los niveles, que manejará el usuario una vez comience el juego. Con este personaje se podrán realizar las actividades de tala, plantación de trigo y compraventa de objetos, así como la interacción con otros personajes y objetos presentes en el videojuego. Este personaje tendrá que cumplir una serie de objetivos en cada nivel para poder pasar al siguiente.



Figura 1: Wuan.

- **Don Diego (el abuelo de Wuan):**

Personaje disponible en todos los niveles, que en el primer nivel al interactuar con él dará indicaciones al usuario de cómo proceder con uno de los objetivos a cumplir en esa fase.



Figura 2: Don Diego.

- **Xoel el mercader:**

Personaje que está presente tanto en el segundo nivel como en el tercer nivel del juego. En el segundo nivel, actúa como propietario de una de las tiendas, donde los usuarios tienen la posibilidad de vender materiales como madera y trigo. Sin embargo, en el tercer nivel, Xoel toma un papel antagonista en la historia. Este personaje implementa un sistema de compraventa en el que ofrece diferentes cantidades de dinero al usuario según el tipo y la cantidad de material ofrecido. En caso de que el usuario no disponga de suficientes unidades en su inventario para satisfacer la cantidad seleccionada para la venta, Xoel no permitirá que la transacción se lleve a cabo.



Figura 3: Xoel el mercader.

- **Eva la modista:**

Personaje disponible en el segundo nivel, que estará en una de las tiendas. Le ofrecerá al usuario la posibilidad de poder comprar diferentes accesorios necesarios para reclutar a los animales a cambio

de dinero. Al igual que el mercader, este personaje seguirá un sistema de compraventa, en el que por cada elemento seleccionado, pedirá a cambio una cantidad de dinero diferente. Este personaje solo dejará al usuario comprar una unidad de cada accesorio, de tal forma que cuando ya se haya comprado un accesorio notificará al usuario de que ya no quedan existencias. Una vez se hayan agotado todos los accesorios el personaje no dejará comprar más accesorios y dirá que tiene que cerrar durante un tiempo.



Figura 4: Eva la modista.

- **Hermanos Pablo y Manuel:**

Personajes disponibles en el primer nivel y segundo nivel. Su único fin es que el usuario pueda interactuar con ellos.



Figura 5: Hermanos Pablo y Manuel.

- **Jordi el obrero:**

Personaje disponible en el primer nivel, al que el usuario tendrá que entregarle diferentes cantidades de dinero y madera a cambio de poder reconstruir la granja.



Figura 6: Jordi el obrero.

- **Gallina Daniel:**

Animal disponible tanto en el segundo nivel como en el tercer nivel, y se encuentra ubicado en la zona de la playa en el segundo nivel, mientras que en el tercer nivel estará en su corral. En el segundo nivel, el usuario puede interactuar con Gallina Daniel, este le solicitará unas zapatillas Jordan a cambio de reclutarlo. En caso de ser el último animal reclutado, le dará al usuario una llave, necesaria en el tercer nivel.



Figura 7: Gallina Daniel.

- **Oveja Óscar:**

Animal disponible en el segundo nivel y que estará situado en la zona del cementerio. En el segundo nivel, el usuario puede interactuar con Oveja Óscar, este le pedirá al usuario una bufanda y una boina a cambio de poder ser reclutado. En caso de ser el último animal reclutado, le dará al usuario una llave, necesaria en el tercer nivel. Además, en el tercer nivel , el animal estará presente en su establo.



Figura 8: Oveja Óscar.

- **Vaca Klara:**

Animal disponible en el segundo nivel y que estará situado en la zona del parking. En el segundo nivel, el usuario puede interactuar con vaca Klara, esta le pedirá al usuario unas gafas Swarovski y una cadena a cambio de poder ser reclutada. En caso de ser el último animal reclutado, le dará al usuario una llave, necesaria en el tercer nivel. Además, en el tercer nivel , el animal estará presente en su establo.



Figura 9: Vaca Klara.

- **Cabra Fer:**

Animal disponible en la mazmorra del tercer nivel. El usuario podrá interactuar con este animal.



Figura 10: Cabra Fer.

### 1.3 Herramientas.

- **Hacha:**

Herramienta utilizada para la tala de árboles.



Figura 11: Hacha.

- **Azada:**

Herramienta utilizada para arar el terreno.



Figura 12: Azada.

- **Regadera:**

Herramienta utilizada para regar el trigo que se ha plantado.



Figura 13: Regadera.

- **Bolsa de semillas:**

Bolsa que contiene las semillas que podrán ser plantadas una vez se haya arado el terreno. Después, para que el trigo crezca, serán regadas con la herramienta de la regadera.



Figura 14: Bolsa de semillas.

## 1.4 Objetos.

- **Madera:**

Material que se podrá recolectar en la tala de árboles en todos los niveles. Por cada tala de árbol se añadirá al inventario cinco unidades de madera.



Figura 15: Madera.

- **Trigo:**

Material que se podrá recolectar durante la plantación de semillas en el terreno arado y regado en el segundo nivel. Por cada sección arada, plantada y regada se añadirá al inventario una unidad de trigo una vez el personaje colisione con el trigo y este haya terminado todas las fases de crecimiento.



Figura 16: Trigo.

- **Bolsa de dinero:**

Objeto que se encontrará disponible inicialmente en la casa de Don Diego en el primer nivel. En este primer nivel, la bolsa de dinero contendrá 100 unidades de dinero. En el nivel 2, este objeto solo se podrá conseguir durante la venta en la tienda del mercader, que le dará al usuario cierta cantidad de dinero en base a los materiales dados. Se utilizará también en la tienda de la modista para poder comprar los accesorios.



Figura 17: Bolsa de dinero.

- **Bufanda y boina:**

Accesorio que estará disponible en el segundo nivel, en la tienda de la modista, para poder ser comprado. Este accesorio deberá ser entregado posteriormente a la oveja Oscar para poder reclutarla. Solo habrá una unidad disponible en la tienda de la modista.



Figura 18: Bufanda y boina.

- **Jordan:**

Accesorio que estará disponible en el segundo nivel, en la tienda de la modista, para poder ser comprado. Este accesorio deberá ser entregado posteriormente a la gallina Daniel para poder reclutarla. Solo habrá una unidad disponible en la tienda de la modista.



Figura 19: Jordan.

- **Gafas y cadena:**

Accesorio que estará disponible en el segundo nivel, en la tienda de la modista, para poder ser comprado. Este accesorio deberá ser entregado posteriormente a la vaca Klara para poder reclutarla. Solo habrá una unidad disponible en la tienda de la modista.



Figura 20: Gafas y cadena.

- **Llave magistral:**

Objeto que le dará el último animal reclutado al usuario en el segundo nivel, con la condición de que todos los animales hayan sido reclutados. Esta llave se añadirá al inventario del jugador y será utilizada en el tercer nivel en la puerta que da acceso a la mazmorra.



Figura 21: Llave magistral.

## 1.5 Diseño.

En el videojuego se establecen tres niveles diferentes (Verano, Otoño e Invierno), y para poder pasar de una fase a otra el jugador tendrá que cumplir con una serie de objetivos que se irán marcando como completados a medida que el usuario los vaya realizando. Estos objetivos los podrá visualizar el usuario, pudiendo ser consciente de cuales han sido marcados como completados y cuáles están aún pendientes por realizar. La mecánica que se establece en el juego es la de ir completando estos objetivos para poder pasar a la siguiente fase, al mismo tiempo que durante estas fases se tendrá que interactuar con los personajes para poder darles los objetos que piden, y que solo se podrán conseguir mediante el sistema de compraventa de las tiendas.

Los objetivos a cumplir por nivel son los siguientes:

- **Nivel 1 (Verano):** Se establecen como objetivos para pasar a la siguiente fase hablar con Don Diego, hablar con Jordi el obrero, talar un árbol, conseguir dinero, darle a Jordi los objetos que solicita y finalmente irse a dormir.
- **Nivel 2 (Otoño):** Se establecen como objetivos para pasar a la siguiente fase arar la tierra, plantar trigo, recoger trigo, hablar con Xoel el mercader y Eva la modista, hablar con todos los animales presentes en ese nivel, darles a los animales los objetos que solicitan y irse a dormir.
- **Nivel 3 (Invierno):** Se establecen como objetivos para terminar el juego el utilizar la llave para abrir la puerta que te dirige a la mazmorra, completar el puzzle de la mazmorra y finalmente hablar con la cabra Fer.

## 2 Escenas.

- **Metodología:** El desarrollo de este videojuego ha seguido una metodología incremental. Para organizarnos hemos usado la herramienta [Asana](#), un software de planificación de proyectos que nos ha permitido repartir las tareas eficientemente. Al inicio de cada semana, asignamos responsabilidades y tareas específicas a cada miembro del equipo para la semana siguiente. Al concluir la semana, consolidamos todo el trabajo realizado y volvemos a distribuir las responsabilidades para la próxima etapa del desarrollo. Este enfoque nos ha permitido mantener un progreso constante y adaptarnos eficientemente a los cambios y desafíos que han surgido durante el proceso de creación del juego.

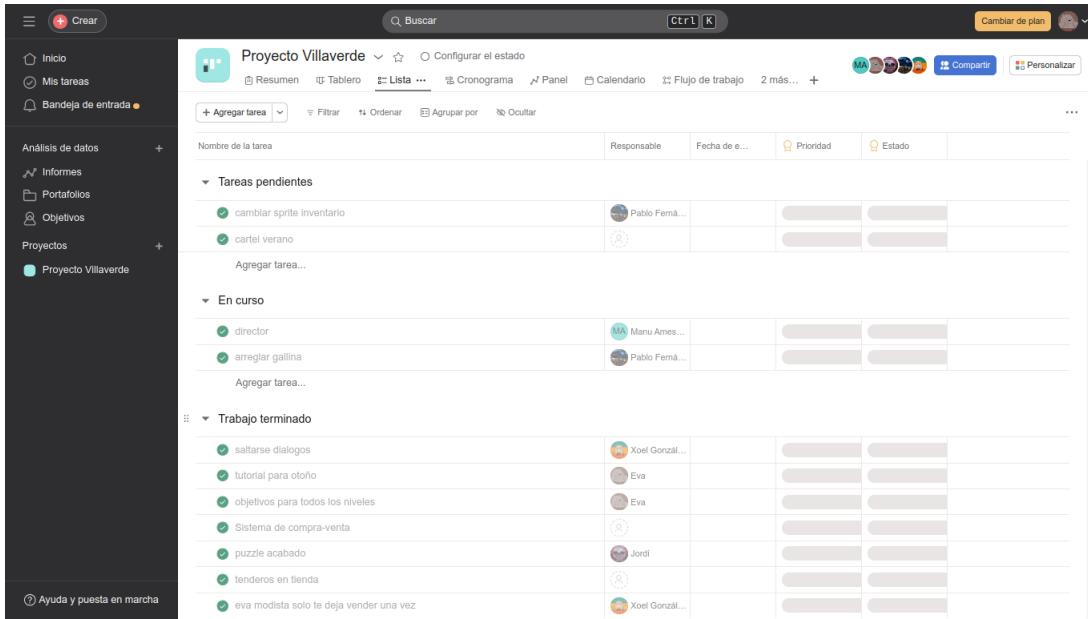


Figura 22: Captura Asana.

El trabajo de los miembros del equipo se dividió de la siguiente manera a lo largo de las semanas:

- José Manuel Amestoy López  
Planificación, sistema de tala, layers, animaciones, cambio de escenas, implementación cinematográficas, interactivas y animales, seguimiento de la cámara y zoom, pantalla de controles, animación de transición.
- Xoel González Pereira  
Inventory, NPCs, todos los diálogos, sistema de compra-venta, interacción con personajes y animales.
- Eva María Arce Ale  
Acciones del personaje, recorte de los sprites, tutorial inicial y tutorial otoño, objetivos de todos los niveles, cambio de nivel.
- Jordi Núñez Arias  
Overlay herramientas, sistema de cultivo (arar, plantar y regar), puzzle final del juego, seguimiento de la cámara al jugador.
- Pablo Fernández Pérez  
Sprites de los personajes, sprites de los accesorios y elementos de interfaz, diseño de mapas, colisiones, menú principal, cinematografía inicial, cambio de escenarios, ocultación de objetos/personajes no presentes en el escenario actual, seguimiento de la cámara al jugador.

- **Descripción global escenas:**

Existen 3 tipos de escena en nuestro juego, las escenas de menú, las escenas de cinamática y las de nivel, hay una de menú que se limita a mostrar un menú con opciones a modificar, las escenas de cinemática, en las que únicamente se reproduce un video y las escenas de nivel, cada una con sus propios parámetros que determinan el mapa, la posición del jugador, los recolectables, animales, NPCs e interactuables disponibles.

- **Transición entre escenas:**

Cada escena tiene una forma diferente de finalizar, el menú termina al empezar a jugar o al salir del juego, las cinemáticas terminan cuando termina el video, planteamos la opción de saltarlas pero al final no la implementamos. Finalmente las escenas de nivel terminan cuando el jugador completa todos los objetivos y va a dormir a la cama. En nuestro caso el juego se desarrolla en un camino completamente lineal, el jugador puede completar los objetivos en el orden que quiera pero el orden de las escenas va a ser siempre el mismo, es un juego en el que no se puede perder ni volver a la escena anterior ni volver al menu principal, por ello tomamos la decisión de no implementar una pila de escenas por falta de tiempo y en su lugar añadimos todas manualmente, cambiando de escena cada vez que una termina.

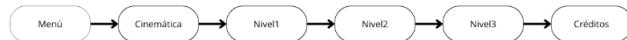


Figura 23: Diagrama de flujo.

## 2.1 Escena 1: Menú.

- **Descripción**

La primera escena es el menú principal del juego, muestra varias opciones, que se detallarán posteriormente, y sirve de introducción al juego.

- **Modelo**

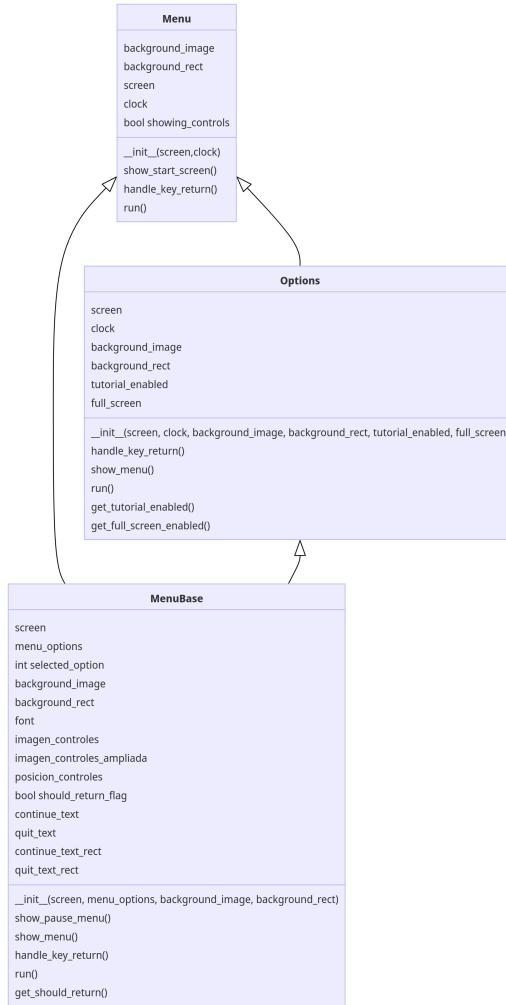


Figura 24: Diagrama de clases del menú.

A continuación destacaremos los patrones de diseño utilizados en esta escena:

- Patrón de Diseño de Comportamiento - *State* (Estado): Este patrón se utiliza para representar los diferentes estados de un objeto y cómo cambia su comportamiento en función de su estado interno. Por ejemplo, en la clase `Menu`, el atributo `showing_controls` actúa como un estado que controla si se están mostrando los controles o no. Esto se maneja a través de la lógica condicional en los métodos `run()` y `handle_key_return()`.
- Patrón de Diseño de Comportamiento - *Template Method* (Método Plantilla): En la clase `MenuBase`, el método `run()` actúa como un método plantilla que define el esqueleto de un algoritmo, pero delega algunos pasos a las subclases (`Menu` y `Options`) para su implementación. Las subclases pueden entonces proporcionar implementaciones concretas de los pasos específicos del algoritmo (`show_menu()`, `show_start_screen()`, `handle_key_return()`, etc.).

- **Detalles de implementación**

A continuación se muestra la implementación del método más importante del archivo menu.py. Este fragmento se encarga de gestionar los eventos de teclado, como las teclas de flecha para navegar por las opciones del menú y la tecla *Enter* para confirmar una selección. Además, maneja la transición entre el menú principal, el menú de opciones y la visualización de los controles del juego. El bucle principal garantiza que el menú se actualice y se muestre correctamente en la pantalla del juego, mientras que la clase MenuBase proporciona funcionalidades compartidas entre los diferentes tipos de menú. Este código es esencial para la interacción del jugador con el juego y la configuración de sus preferencias antes de comenzar a jugar.

```

def run(self):
    # Método para ejecutar el menú
    self.in_menu = True
    self.options = None
    self.tutorial_enabled = True
    self.full_screen = False
    while self.in_menu:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
            elif event.type == pygame.KEYDOWN:
                if event.key == pygame.K_DOWN:
                    self.selected_option = (self.selected_option + 1) % len(self.menu_options)
                elif event.key == pygame.K_UP:
                    self.selected_option = (self.selected_option - 1) % len(self.menu_options)
                elif event.key == pygame.K_RETURN:
                    self.handle_key_return()
            if self.options:
                self.options.run()
                self.tutorial_enabled = self.options.get_tutorial_enabled()
                self.full_screen = self.options.get_full_screen_enabled()
                if self.options.get_should_return():
                    self.options = None
            if self.showing_controls:
                self.screen.blit(self.imagen_controles_ampliada, self.posicion_controles)
                pygame.display.flip()
                while True:
                    event = pygame.event.wait()
                    if event.type == pygame.KEYDOWN and event.key == pygame.K_RETURN:
                        self.showing_controls = False
                        break
            else:
                self.show_menu()

    self.clock.tick(FPS)

```

## 2.2 Escena 2: Cinemática.

- **Descripción**

La segunda escena es una cinemática inicial que explica la historia del juego y los sucesos de la trama transcurridos antes de la llegada de Wuan, contextualiza al juego y enriquece la historia del mismo.

- **Modelo**

Video
path
name
_video
int_frame_num
frame_rate
frame_count
frame_delay
duration
original_size
current_size
bool active
frame_surf
alt_resize
__init__(path)
close()
restart()
set_size(size)
set_volume(volume)
get_volume()
get_paused()
pause()
resume()
get_pos()
toggle_pause()
_update()
seek(seek_time)
draw(surf, pos, force_draw)

Figura 25: Clase Video.

- **Detalles de implementación**

Para la implementación de las cinemáticas nos hemos apoyado en el script pyviplayer.py que proporciona facilidades para introducir videos en pygame, la cinemática de intro es simplemente un video adaptado al tamaño de la ventana, como se ve en el código:

```
def playIntro(self):  
    self.video_path = os.path.join(self.current_path, self.intro_name)  
    self.intro = Video(self.video_path)  
    self.intro.set_size((SCREEN_WIDTH, SCREEN_HEIGHT))  
    while self.intro.active:  
        self.intro.draw(self.screen, (0,0))  
        pygame.display.update()
```

## 2.3 Escena 3: Verano

- Descripción

La escena de verano se caracteriza por tres cosas, el mapa está en ruinas, únicamente se puede hablar con Jordi y Don Diego y se aprende a talar y recoger objetos, es el mapa más colorido.

- Modelo

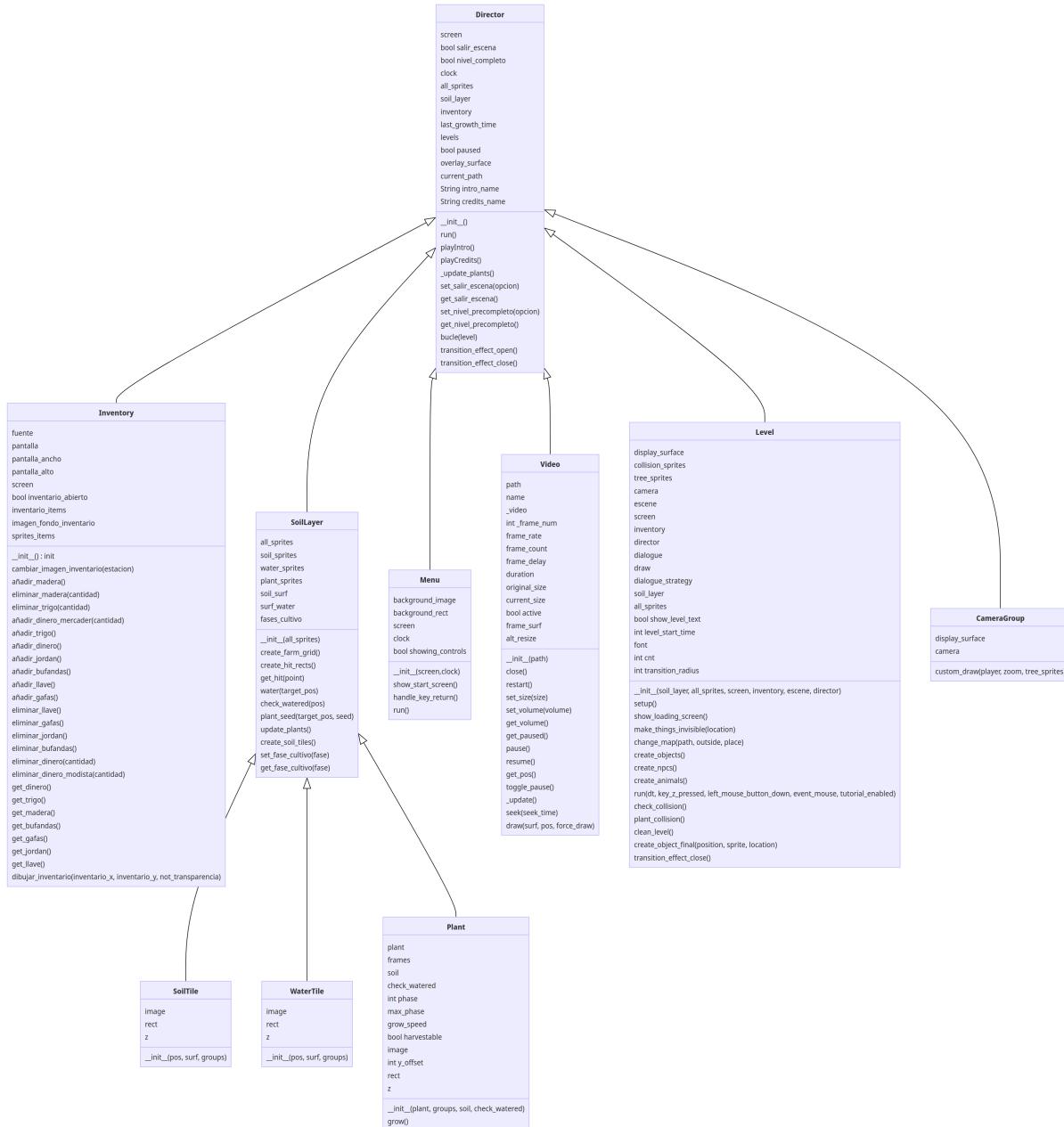


Figura 26: Clase Director.

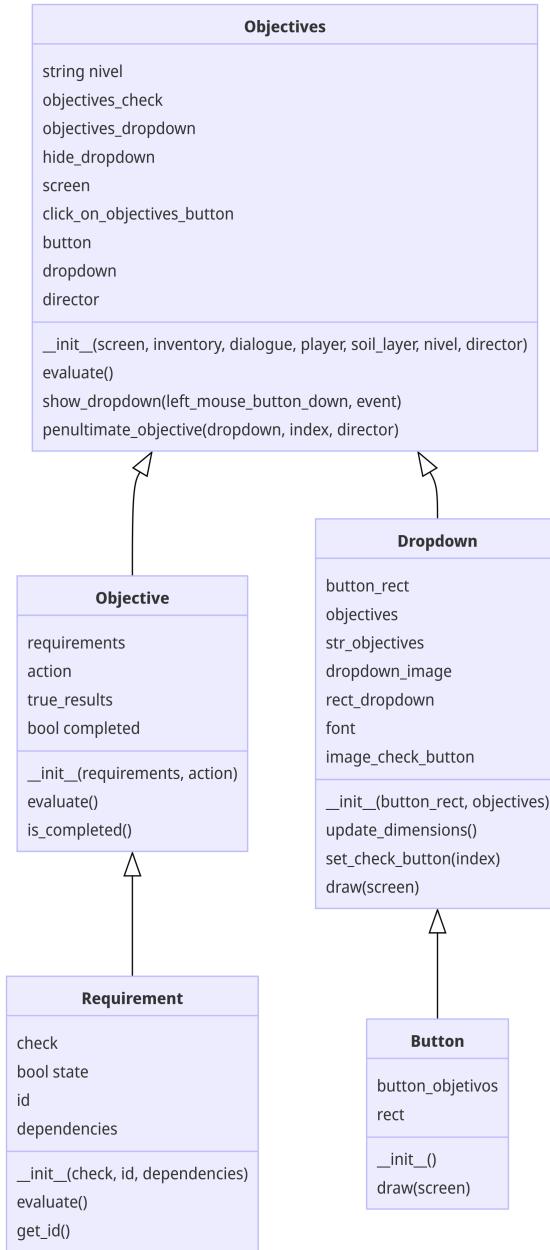


Figura 27: Clase Objectives.

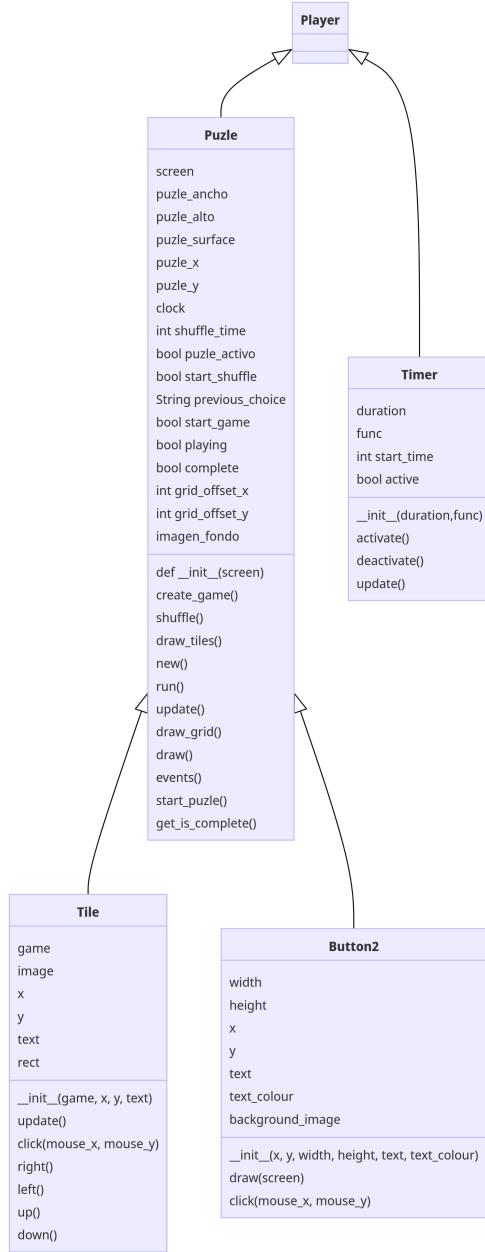


Figura 28: Clase Player.

Los diagramas correspondientes a la estrategia de los diálogos y el nivel se encuentran en la carpeta diagramas ya que debido a su tamaño no se pueden incluir de forma adecuada en la memoria.

En este caso en el nivel 1 tenemos varios patrones de diseño, los de player, level y diálogos están incluídos más abajo porque son comunes en todos los niveles, como los árboles son la mecánica principal del nivel 1 los definiremos aquí. Los árboles usan dos patrones de diseño, una máquina de estados que indica si el árbol está vivo o no, cambiando así su sprite por el de un tocón para indicar que ya está talado, y usan también el patrón observador para entregar al jugador la madera correspondiente una vez está talado, esto la hace usando una variable externa que indica la muerte de un árbol para proporcionar madera al inventario.

- **Detalles de implementación**

El único sistema nuevo a parte de level y diálogos en el nivel 1 es el sistema de tala, aquí incluimos

el código para gestionar el cambio de *sprite* del árbol por un tocón y la interacción con el mismo, la peculiaridad de este sistema es el tipo de tronco que se elige al talar un árbol para que su tocón se corresponda:

```

class Tree(Generic):
    def __init__(self, pos, surf, groups, name, inventory, season=0):
        super().__init__(pos, surf, groups)

        self.rect.inflate_ip(20, 20)
        self.hitbox.midleft = self.rect.midleft
        self.visible = True
        self.name = name
        self.inventory = inventory
        self.visible_image = surf # Almacena la imagen original
        if season == 0:
            self.stump_surf = pygame.image.load(f'code/sprites/ambiente/ambiente_primavera/
                {"tronco1" if name == "arbol1" else ("tronco2" if name == "arbol2" else
                "tronco3")}.png').convert_alpha()
        elif season == 1:
            self.stump_surf = pygame.image.load(f'code/sprites/ambiente/ambiente_verano/
                {"tronco1" if name == "arbol1" else ("tronco2" if name == "arbol2" else
                "tronco3")}.png').convert_alpha()
        elif season == 2:
            self.stump_surf = pygame.image.load(f'code/sprites/ambiente/ambiente_otoño/
                {"tronco1" if name == "arbol1" else ("tronco2" if name == "arbol2" else
                "tronco3")}.png').convert_alpha()
        else:
            self.stump_surf = pygame.image.load(f'code/sprites/ambiente/ambiente_invierno/
                {"tronco1" if name == "arbol1" else ("tronco2" if name == "arbol2" else
                "tronco3")}.png').convert_alpha()
        self.alive = True
        self.health = 1

    def damage(self):
        if self.visible:
            self.health -= 1

    def check_death(self):
        if self.visible and self.alive:
            if self.health <= 0:
                # Ajustar posición inicial del sprite
                self.rect.y += 20
                #self.rect.x -= 20
                self.image = self.stump_surf
                self.visible_image = self.stump_surf

                # Ajustar el rectángulo para el tocón
                self.rect = self.image.get_rect(midbottom=self.rect.midbottom)
                self.hitbox = self.rect.copy()
                self.alive = False

```

```
    self.inventory.añadir_madera()
```

## 2.4 Escena 4: Otoño

- **Descripción**

La escena de otoño es la más larga del juego, se abren las tiendas, se desbloquea el sistema de cultivo, se abren los caminos a las zonas de los animales y el jugador tiene que completar todos los objetivos correspondientes a este nivel.

- **Modelo**

De nuevo los patrones de level, dialogos y player están más abajo, para este nivel tambien se incluyen los de los animales, igual que en el nivel 1 en el nivel 2 se desbloquea un nuevo sistema, el sistema de cultivo, el cual se construye usando los siguientes patrones:

- Patrón Observador:

El método *\_update\_plants* se encarga de actualizar el crecimiento de las plantas. Esto es similar al patrón Observador, donde los objetos (plantas) notifican a otros objetos (en este caso, la clase *SoilLayer*) sobre un cambio en su estado.

- Patrón *Factory* (Factoría) y Patrón *Builder* (Implícitos):

La clase *SoilLayer* actúa como una fábrica para crear instancias de *SoilTile*, *WaterTile*, y *Plant*. Además, en el método *create\_soil\_tiles*, se construyen los objetos *SoilTile* y se añaden a *self.soil\_sprites*, lo que se asemeja al patrón *Builder*.

- Patrón *Strategy* (Implícito): El método *grow* en la clase *Plant* utiliza una estrategia para determinar cómo debe crecer la planta. Esto podría ser considerado una forma de implementar el patrón *Strategy*, donde se encapsulan algoritmos específicos y se pueden intercambiar dinámicamente.

- Patrón *State* (Estado): Los atributos fases cultivo en la clase *SoilLayer* representan el estado de la capa del suelo en términos de diferentes fases de cultivo (arar, sembrar, regar). Estos estados afectan el comportamiento del objeto, lo que se asemeja al patrón *State*.

- **Detalles de implementación**

El código que controla la velocidad de crecimiento de las plantas está en director y funciona con tiempo real en lugar de *delta time*:

```
def _update_plants(self):
    current_time = time.time()
    elapsed_time = current_time - self.last_growth_time

    if elapsed_time >= 5:
        self.soil_layer.update_plants()
        self.last_growth_time = current_time
```

## 2.5 Escena 5: Invierno

- **Descripción**

La escena de invierno conserva el sistema de tala pero a estas alturas ya no sirve de nada, pues las tiendas cierran y no hace falta la madera, es un escenario prácticamente vacío en interacciones que sirve de transición hasta la mazmorra en la que se encuentra el puzzle final del juego.

- **Modelo**

En el nivel 3 todos los patrones que se usan se mencionan en los anteriores niveles o en el apartado de patrones, a excepción del puzzle, el cual usa un patrón *Strategy*, modificando su comportamiento en tiempo real dentro del juego, usa tambien un patrón observador para indicar cuando se acaba el puzzle y se desbloquea la sala final del videojuego.

- **Detalles de implementación**

La novedad de este nivel es el puzzle, el cual se desordena solo y tiene este código:

```

def create_game(self):
    grid = [[x + y * GAME_SIZE for x in range(1, GAME_SIZE + 1)]
            for y in range(GAME_SIZE)]
    grid[-1][-1] = 0
    return grid

def shuffle(self):
    possible_moves = []
    for row, tiles in enumerate(self.tiles):
        for col, tile in enumerate(tiles):
            if tile.text == "empty":
                if tile.right():
                    possible_moves.append("right")
                if tile.left():
                    possible_moves.append("left")
                if tile.up():
                    possible_moves.append("up")
                if tile.down():
                    possible_moves.append("down")
                break
            if len(possible_moves) > 0:
                break

    if self.previous_choice == "right":
        possible_moves.remove("left") if "left" in possible_moves else possible_moves
    elif self.previous_choice == "left":
        possible_moves.remove("right") if "right" in possible_moves else possible_moves
    elif self.previous_choice == "up":
        possible_moves.remove("down") if "down" in possible_moves else possible_moves
    elif self.previous_choice == "down":
        possible_moves.remove("up") if "up" in possible_moves else possible_moves

    choice = random.choice(possible_moves)
    self.previous_choice = choice
    if choice == "right":
        self.tiles_grid[row][col],
        self.tiles_grid[row][col + 1] = self.tiles_grid[row][col + 1], \
                                         self.tiles_grid[row][col]
    elif choice == "left":
        self.tiles_grid[row][col],
        self.tiles_grid[row][col - 1] = self.tiles_grid[row][col - 1], \
                                         self.tiles_grid[row][col]
```

```

        self.tiles_grid[row][col]

    elif choice == "up":
        self.tiles_grid[row][col],
        self.tiles_grid[row - 1][col] = self.tiles_grid[row - 1][col], \
            self.tiles_grid[row][col]

    elif choice == "down":
        self.tiles_grid[row][col],
        self.tiles_grid[row + 1][col] = self.tiles_grid[row + 1][col], \
            self.tiles_grid[row][col]

```

## 2.6 Escena 6: Créditos.

- **Descripción**

Los créditos muestran una pantalla de fin del juego y los diferentes nombres del los miembros del equipo y agradecimientos.

- **Modelo**

Igual que escena 2.

- **Detalles de implementación**

Para el video de los créditos hacemos lo mismo que en la introducción pero con otro video.

## 2.7 Patrones de diseño:

- Patrón *Singleton* en Director, la clase director presenta un patrón *singleton* ya que solo puede haber una instancia de director, esta instancia tiene acceso global, un constructor privado y presenta control sobre la creación para evitar mas instancias.
- Patrón Observador en los niveles, todos los niveles usan el patrón observador, en el bucle principal del juego se espera al cambio de valor de la variable salir\_escena para interrumpir el bucle y salir de la escena actual, concluyendo así el nivel.
- Patrón plantilla en NPCs, Animales e Interactuables, todos estos objetos utilizan el patrón plantilla, se basan en una clase general y cada uno tiene sus propios atributos, ya sea la ruta a sprites, el movimiento el diálogo asociado o la posicion.
- Patrón *Strategy* en el *Player*, este patrón permite cambiar dinámicamente el comportamiento de una clase en tiempo de ejecución, en este caso del jugador, permitiendo su movimiento, también usamos este patrón para todos los diálogos, tanto de personajes como de animales.
- Maquinas de estados en Animales y *Player*, *Player* tiene diferentes estados, inactivo en todas las direcciones, caminando en todas las direcciones y usando las herramientas en todas las direcciones, por otro lado los animales tienen también diferentes estados, pueden estar en estado *prime* o no, esto indica si llevan puesto su accesorio o no, por otro lado pueden estar en estado inactivo o caminando, todos los *sprites* para representar los diferentes estados se eligen gracias a su máquina de estados, además también tenemos una máquina de estados en nivel, la cual decide el mapa, las posiciones y los elementos dependiendo del nivel.
- Patrón *Factory*, el método *create\_objects* dentro de la clase *Level* implementa un patrón *Factory*, ya que crea y devuelve una lista de objetos (*InteractableObject* y *NPC*) basados en la escena del juego, lista que usamos para ocultar o mostrar los objetos cuando cambiamos de mapa.

- Patrón observador: Se utiliza para el cambio de inventario en función de la estación/nivel. El observador (*Inventory*) es notificado cuando el sujeto observable (*Level*) cambia su estado, a través del método 'cambiar\_imagen\_inventario' de *Inventory*.

### 3 Aspectos destacables.

Cabe mencionar que algunos de los *sprites* utilizados en este juego se han creado especialmente para el mismo. Esto incluye tanto a los *sprites* del protagonista, Wuan, como a todos los personajes secundarios. En el caso de los animales, los *sprites* provienen del paquete '*Premium Chill Harvest-Farm*', que hemos adquirido por el precio de diez euros, pero han sido modificados a mano para añadir los distintos accesorios.



Figura 29: Acciones de Wuan.

Con lo que respecta a los mapas, se han utilizado los *tiles* del paquete anterior y de '*Super Retro World Interior Pack*', en este caso, gratuito. Aún así, los mapas han sido diseñados empleando *Tiled*. De igual forma otros *sprites* como el inventario o los objetivos, entre otros, han sido creados también expresamente para este juego. Los *sprites* de los personajes en los diálogos han sido creados mediante el uso de IA para que se ajustasen a nuestros requisitos y han sido modificados a mano para solucionar algún error.



Figura 30: Mapa de otoño.

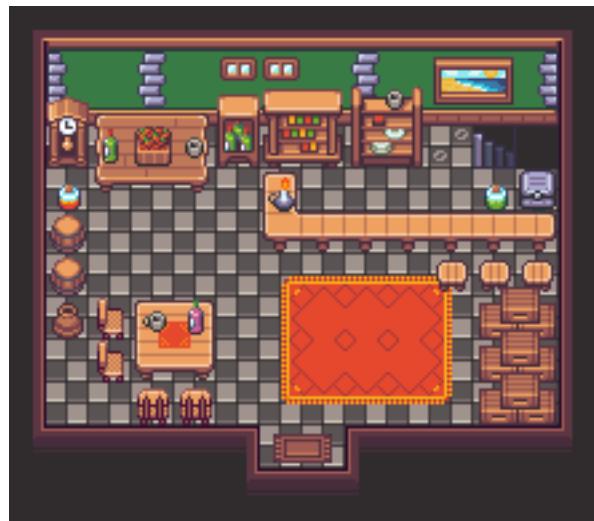


Figura 31: Tienda de Xoel.

La cinemática inicial y los créditos, también han sido creados expresamente para este videojuego. Finalmente podemos destacar el menú de pausa (Figura 32), se accede presionando la tecla 'ESC' y pausa el bucle principal del juego para mostrar los controles, sabemos que no es una buena práctica pausar el bucle principal del juego, pero en nuestro caso no afecta a las posiciones de los animales (que son los únicos que se mueven) ni del jugador, pues también los pausamos, por eso decidimos dejarlo así ya que no afecta de ninguna manera a la experiencia de juego.

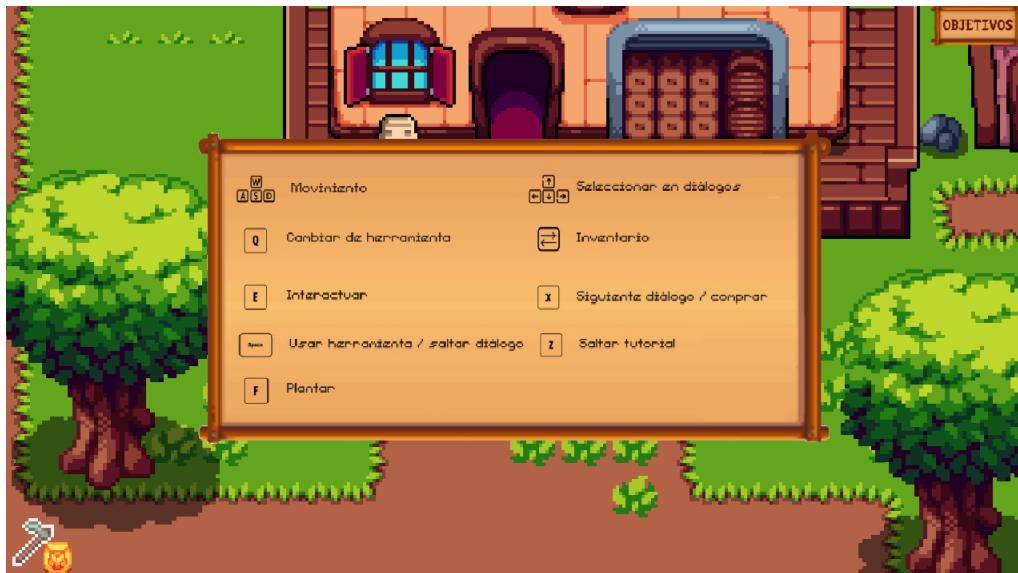


Figura 32: Pantalla de pausa.

## 4 Manual de usuario.

A continuación se explica como realizar las acciones básicas en el videojuego:

### 4.1 Menú principal.

En el menú principal, puedes elegir una opción utilizando las flechas del teclado y presionando Enter.



Figura 33: Menú principal.

En el apartado 'Opciones' se permite tanto Activar/Desactivar el tutorial como Activar/Desactivar el modo pantalla completa.



Figura 34: Opciones.

En el apartado 'Controles' se pueden ver todas las teclas que puedes utilizar en el juego y las acciones asignadas a cada una de ellas.



Figura 35: Controles.

## 4.2 Objetivos.

En cada nivel se pueden ver los objetivos haciendo click en la pestaña 'OBJETIVOS'. Cada vez que se cumpla uno pasará a verse como completado.



Figura 36: Objetivos del nivel.

#### 4.3 Juego pausado.

Se puede pausar el juego en cualquier momento presionando 'Esc', lo que permite a su vez ver los controles del juego.



Figura 37: Juego pausado.

#### 4.4 Interacción con NPCs.

Para hablar con un personaje, el jugador deberá acercarse al mismo y presionar la tecla E, en ese momento se abrirá el diálogo. Puedes completar el diálogo presionando el Espacio y continuar presionando la tecla X.



Figura 38: Ejemplo de diálogo con NPC.

Algunos NPCs solicitarán objetos al jugador (Figura 39). En estos casos se proporcionarán dos opciones de diálogo a elegir. El jugador deberá seleccionar una opción utilizando las flechas del teclado y presionando X.



Figura 39: Ejemplo de interacción con NPC.

#### 4.5 Sistema de tala.

El jugador puede talar usando 'Espacio', tras haber seleccionando como herramienta la azada con la 'Q'. Para talar un árbol es necesario golpearlo desde las distintas posiciones posibles hasta que este caiga.



Figura 40: Acción de talar.



Figura 41: Ejemplo de árbol talado.

#### 4.6 Sistema de cultivo.

Para cultivar el jugador debe situarse en la zona cultivable que se indica en el Nivel 2 y realizar todas las acciones situado justo encima de la porción de tierra en la que deseas plantar.

Una vez en la zona cultivable se puede arar la tierra usando 'Espacio', tras haber seleccionando como herramienta la azada con la 'Q'.



Figura 42: Acción de arar.

Posteriormente se puede usar la bolsa de semillas de trigo para plantar presionando 'F'.



Figura 43: Acción de plantar.

Finalmente puedes regar usando 'Espacio', tras haber seleccionado como herramienta la regadera con la 'Q'.



Figura 44: Acción de regar.

#### 4.7 Sistema de compra-venta.

Una vez el jugador alcanza el Nivel 2, desbloqueará las tiendas de Eva la modista y Xoel el mercader. El jugador puede hablar con Xoel el mercader para vender sus productos, tanto madera como trigo. Para ello, debe seleccionar mediante el uso de las flechas del teclado la cantidad que desea vender de un producto en concreto y confirmar con 'X'.



Figura 45: Venta de objetos.

El jugador puede hablar con Eva la modista para comprar los distintos artículos necesarios para que los animales sean felices y regresen a la granja. Para ello, debe seleccionar mediante el uso de las flechas del teclado el artículo que desea comprar y confirmar con 'X'.



Figura 46: Compra de artículos.

## 4.8 Otras interacciones.

### 4.8.1 Interacción con carteles.

Se puede interactuar con carteles si estás situado cerca de ellos presionando la tecla 'E'.



Figura 47: Interacción con carteles.

### 4.8.2 Interacción con la puerta de la mazmorra.

Cuando llegas al Nivel 3, debes usar la llave obtenida en el Nivel 2 para acceder a la mazmorra final.



Figura 48: Interacción con la puerta de la mazmorra.

## 4.9 Puzzle.

Como reto final, el jugador deberá resolver un puzzle empleando el ratón para mover las piezas. Una vez ha conseguido ordenar las piezas el puzzle se establecerá como completado y se abrirá la puerta, permitiendo al jugador acceder a la sala final y así poder rescatar a la cabra Fer.



Figura 49: Puzzle.

#### 4.10 Librerías que hay que instalar

- pygame
- pytmx
- ffpyleyer
- pymediainfo

## 5 Reporte de bugs.

- **Diálogos:**

Cuando se quiere completar un diálogo con la tecla SPACE para no esperar a que se dibuje entero al mismo tiempo que se quiere avanzar el diálogo con la tecla X, si en ese índice de diálogo se iban a dibujar una serie de opciones, al presionar ambas teclas al mismo tiempo se cierra el diálogo y el jugador se queda bloqueado. Para desbloquear al jugador hay que volver a pulsar la tecla X para volver a abrir el diálogo restante. Este problema puede ser debido al uso de las dos teclas, una para rellenarlo y otra para avanzar al siguiente índice. Una solución que se ha pensado para simplificarlo, sería el uso de una sola tecla tanto para llenar el diálogo si el jugador no quiere esperar a que se dibuje de forma completa como para avanzar al siguiente índice. Esto se ha pensado que se podría realizar mediante una espera que le impida al usuario el realizar las dos acciones al mismo tiempo o poder bloquear de alguna forma una de las acciones.

- **Árboles:**

Los árboles tienen dos bugs, la colisión con el hacha no es perfecta, falla en muchos ángulos pero todos los árboles se pueden talar, esto se debe a que la posición del jugador y el sprite del árbol tienen que colisionar para que se considere que se ha talado, podría arreglarse añadiendo un offset a la posición del jugador a la hora de talar indicando donde se sitúa el hacha, el otro bug respecto a los árboles lo tiene el tocón que queda después de talarlo, al tratarse del mismo objeto que el árbol inicial no se ajusta correctamente la capa respecto al jugador, pudiendo verse con un orden que no corresponde, podría arreglarse redefiniendo el orden en el que se dibujan los tocones respecto al jugador.

- **Pop de los personajes:**

Al cambiar de escenario los personajes desaparecen para evitar que se vean donde no deberían en otros escenarios, el bug es la visible aparición de los personajes en el escenario, pues se ve como de la nada aparecen, esto se debe a la recarga de sus sprites que ocurren después de cargar el mapa, probamos añadiendo una pantalla de carga en paralelo pero no nos dio resultado pues siempre se cargan en el siguiente frame al mapa, así que decidimos dejarlo por falta de tiempo.

- **Cantidad de objetos en el inventario:**

Si el inventario contiene más de cinco objetos, algunos de ellos sobresalen fuera de la imagen del inventario. Esto ocurre porque los objetos que se muestran en el inventario están guardados en una lista sin límite. Para evitar esta situación, se podría establecer un indicador que impida que el tamaño de la lista exceda cinco elementos.

- **Fuente:**

La fuente utilizada no permite vocales acentuadas, ni la letra 'ñ', ni otros símbolos como el inicio de la exclamación e interrogación, entre otros. Esto se podría evitar utilizando otra fuente que admita el español.

- **Hablar con el mercader:**

Si hablas con el mercader, después con la modista y vuelves a interactuar con el mercader, el juego se cierra automáticamente debido a un fallo en el índice de las opciones del menú, en la variable opcion\_menu.