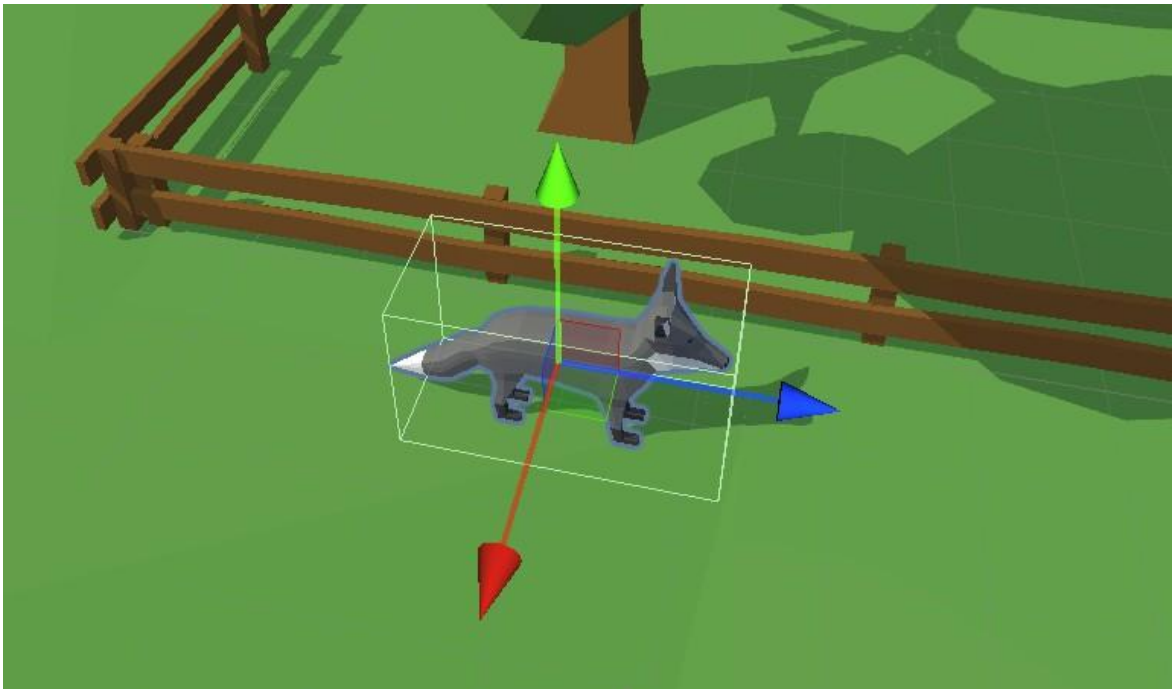
Timmy es un niño lleno de energía y curiosidad, es hijo de un reconocido granjero que ha dedicado su vida al cuidado de la tierra y los animales. Para este año, su padre ha decidido que es hora de que Timmy aprenda lo que significa mantener una granja en perfecto estado. Con la llegada de la primavera, el campo se llena de vida, pero también de tareas importantes para garantizar que todo funcione como debe.
En esta aventura, Timmy tendrá que realizar una serie de actividades esenciales, desde recoger la basura para mantener el ambiente limpio, hasta organizar la leña para el invierno que se avecina. También deberá guiar a los animales a sus respectivos corrales, asegurándose de que todos estén cómodos y bien cuidados. Cada tarea no solo será un desafío, sino también una oportunidad para aprender la importancia del trabajo duro, el respeto por la naturaleza y la responsabilidad.

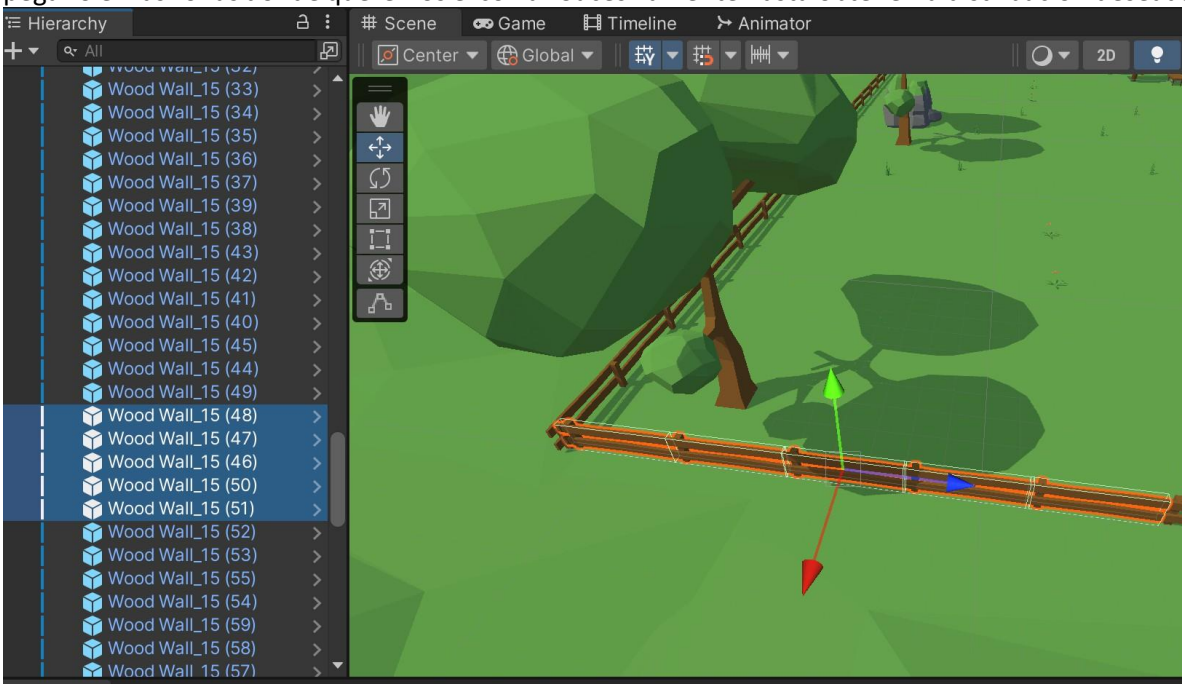
Desarrollo

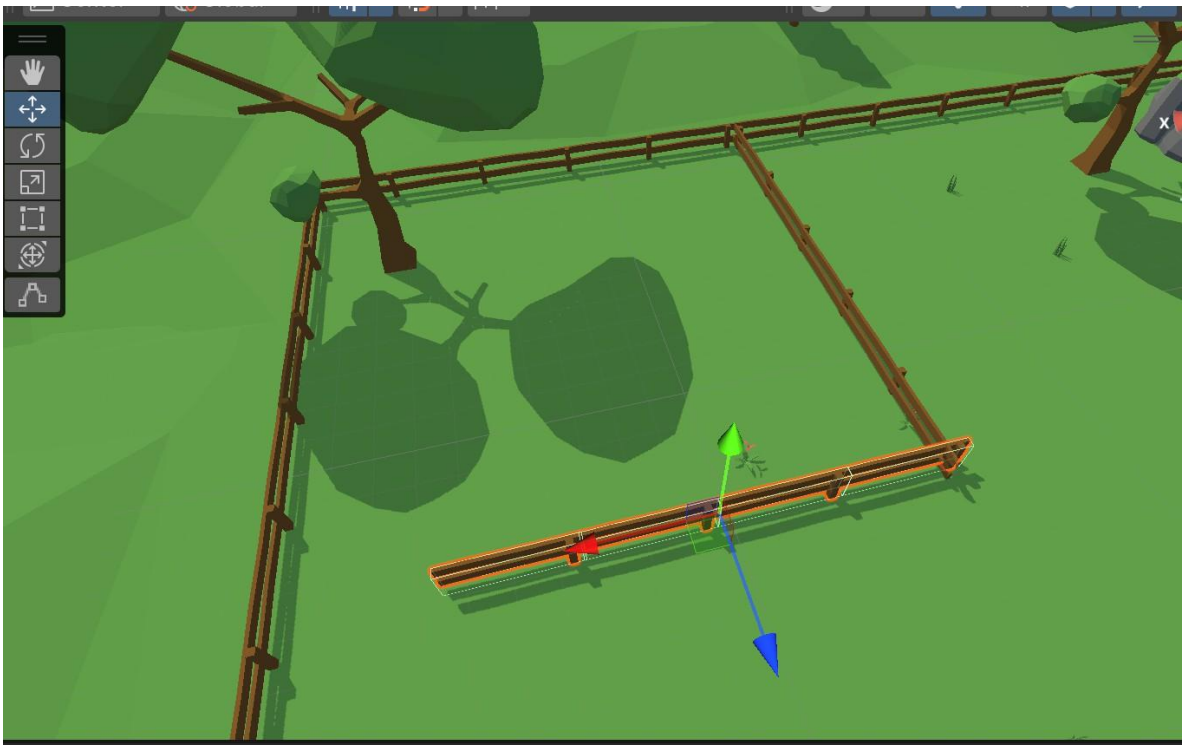
Como primer paso en este proyecto, se ha bajado el mapa que se utilizará como base para comenzar con el entorno virtual, el cual adaptaremos a nuestras necesidades y objetivos.

Comenzando por eliminar algunos objetos innecesarios o irrelevantes.

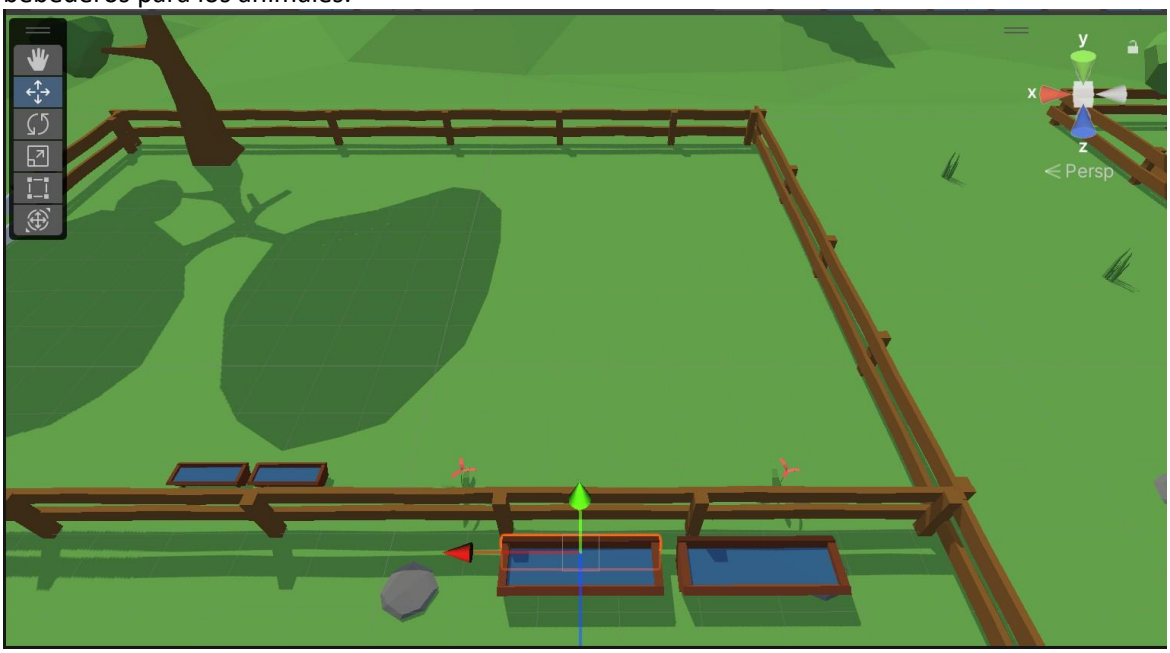


Los que si necesitemos, solamente los movemos a donde queramos para que vaya tomando forma nuestra pequeña granja, por ejemplo para ir creando los corrales, basta con tomar la forma de alguno ya creado y pegarlo en las zonas donde queremos el corral. Sucesivamente hasta obtener la distribución deseada.





Para despues ir agregando algunos elementos decorativos para el entorno, como en este caso flores y bebederos para los animales.



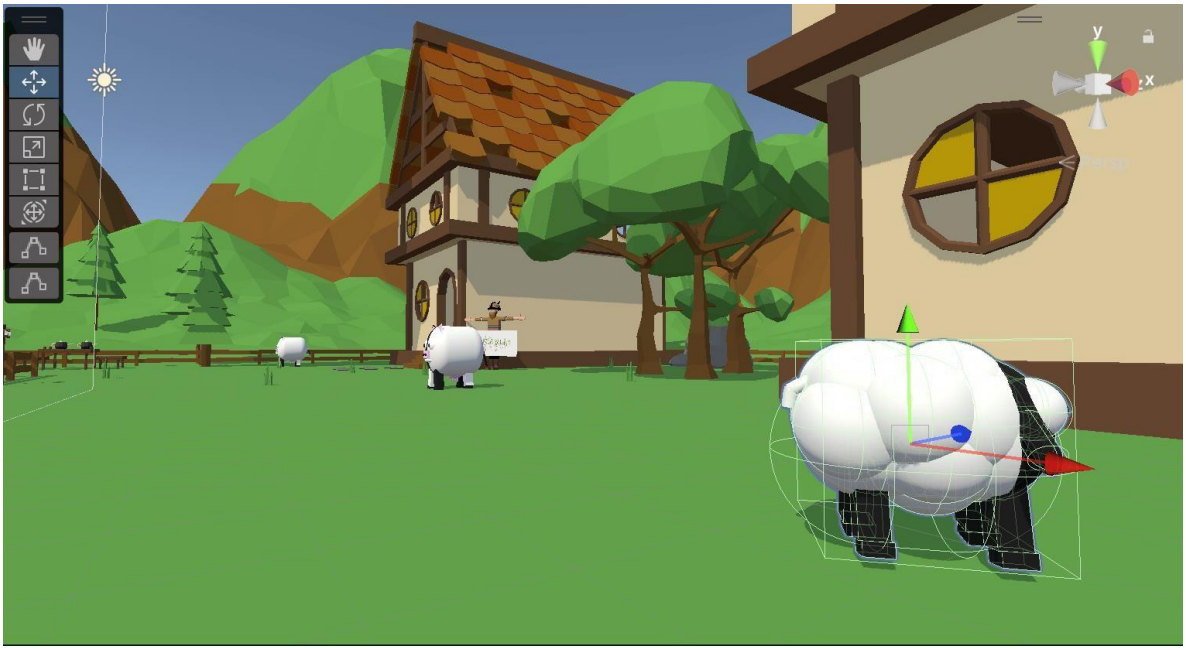
Carteles ilustrativos para identificar a que corral corresponde cada animal

Personajes auxiliares ya que dan avisos, datos informativos o advertencias

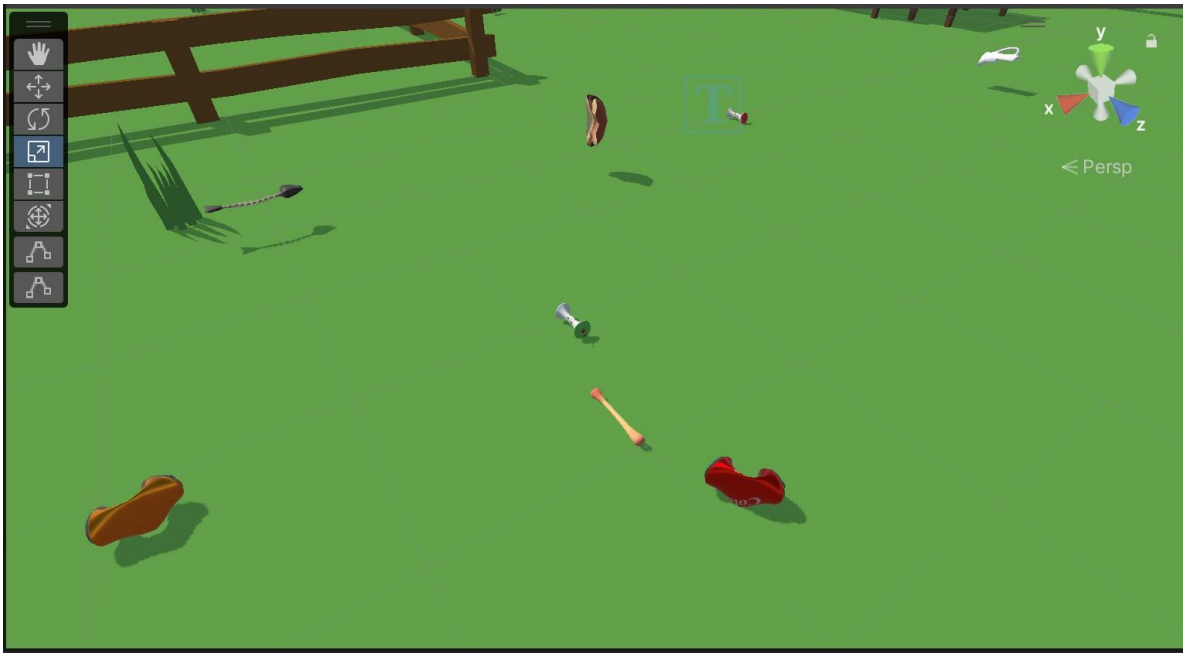
Y algunos carteles con mensajes iconicos

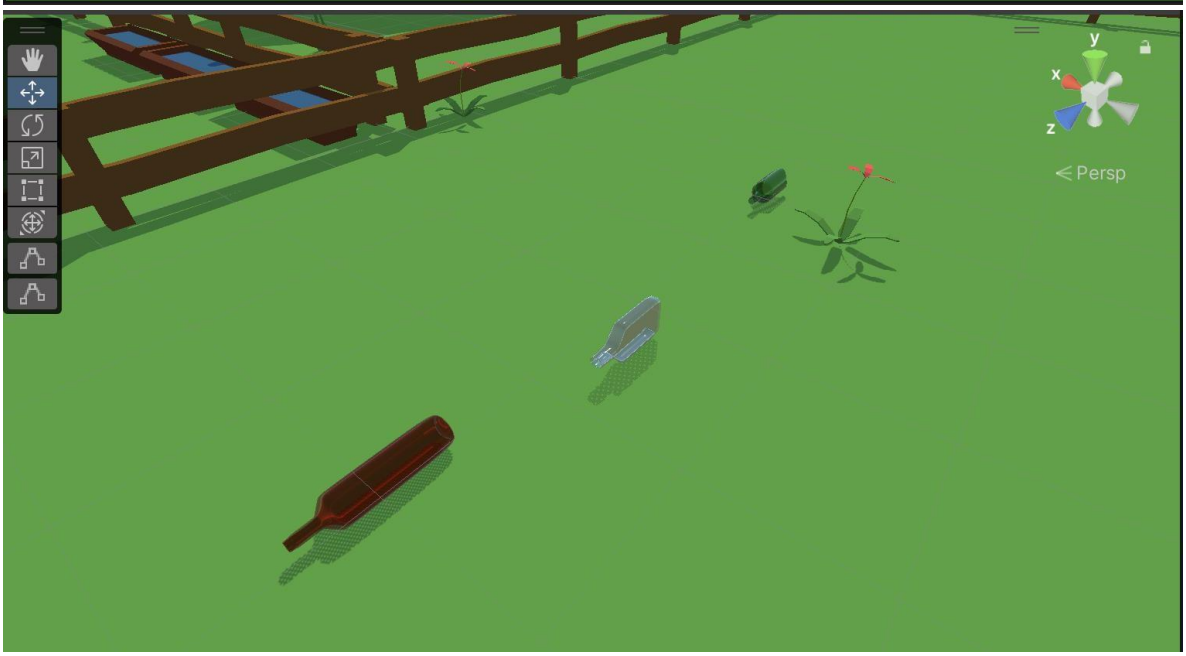
La simulacion del sol mediante directional light

Los animalitos correspondientes

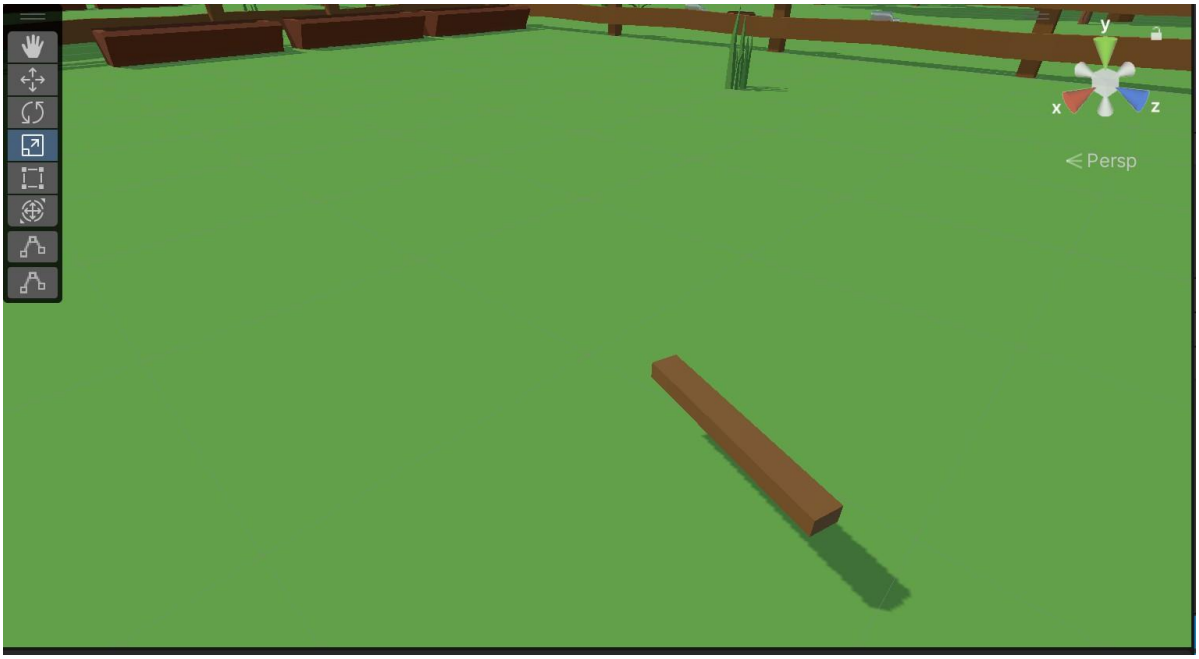


La basura que hay que eliminar (ya que es una de las principales dinamicas)



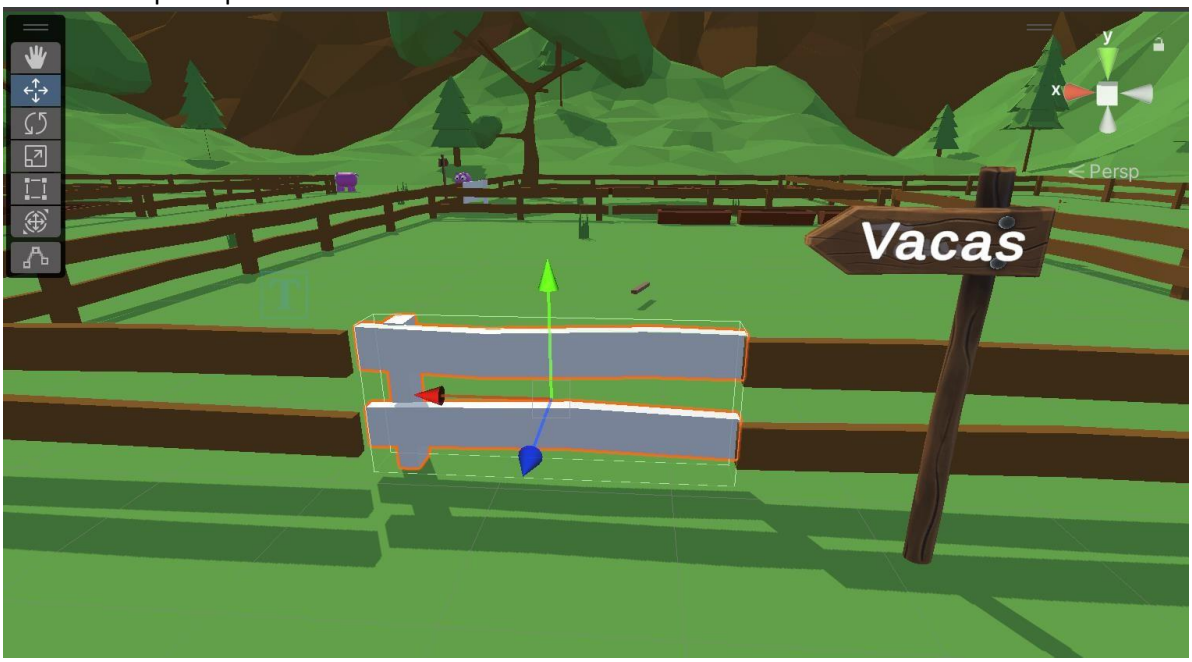


Las varas de madera, que tambien es otra de las dinamicas

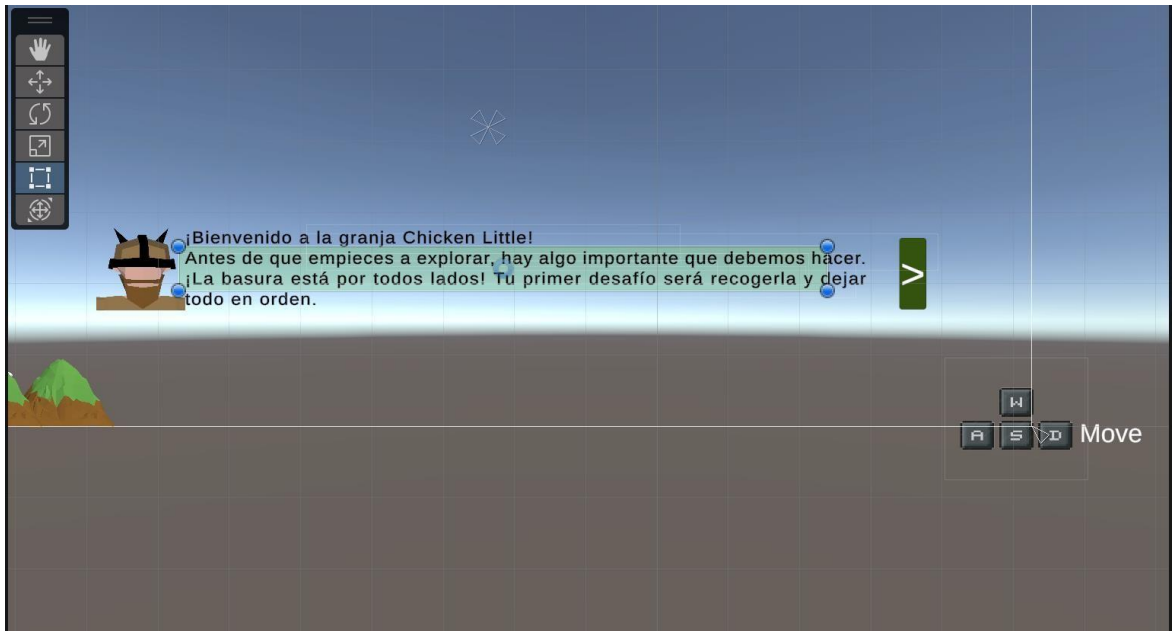


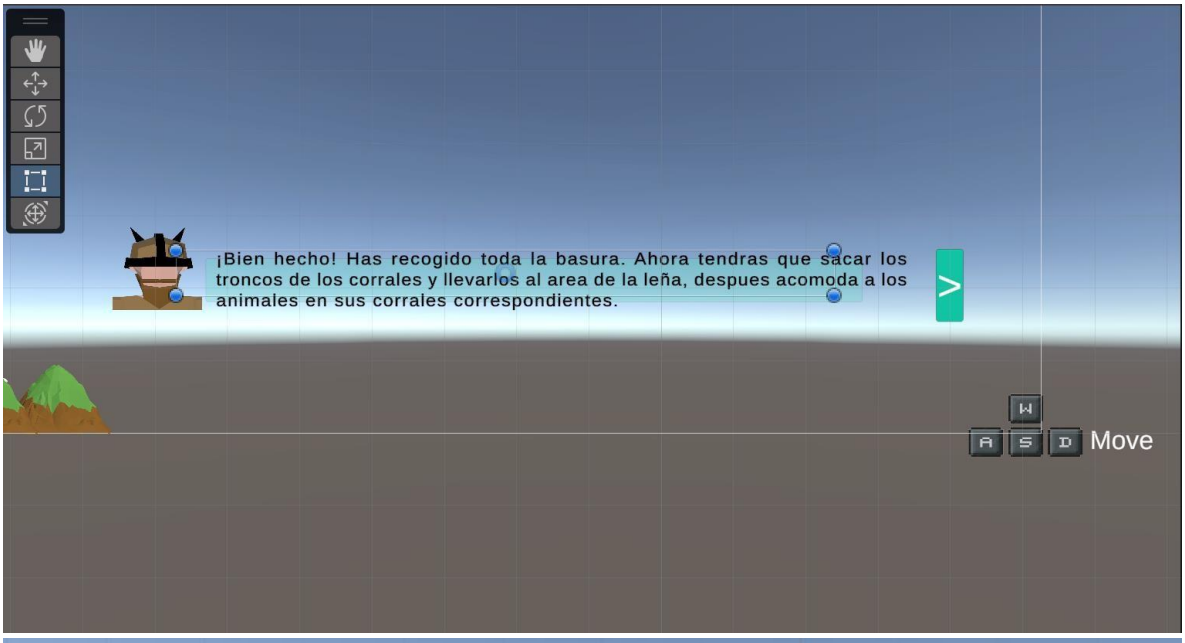
El pequeño Timmy, personaje que nos acompañó a lo largo de nuestras practicas y entregas(al cual se le puso la mainCamera en una zona estrategica detrás de su cabeza para dar una perspectiva de tercera persona)

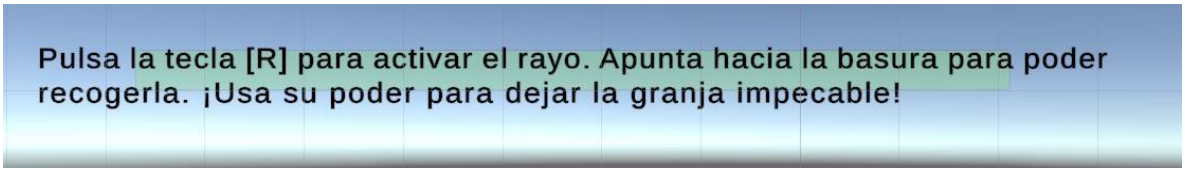
Puertas automaticas (se les añadio un color blanco para destacar cual es la entrada y se que logre diferenciar del resto de las vallas) Las cuales se abren o se cierran mediante el uso del collider, se agrega un objeto en el piso invisible para poder activarle los triggers y al colisionar con este objeto se pueda dar la orden de que la puerta se abrirá.

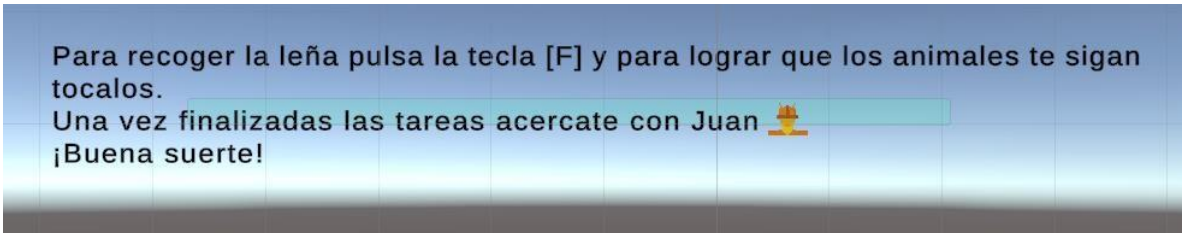


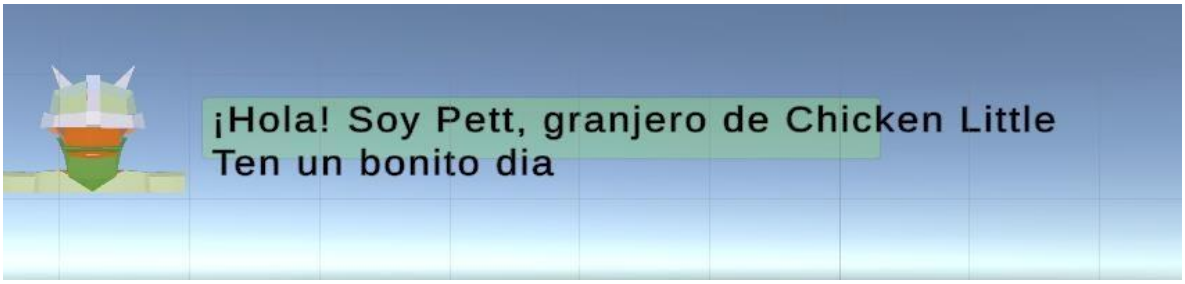
Estos son algunos de los mensajes que se muestran mediante los paneles




¡Bien hecho! Has recogido toda la basura. Ahora tendras que sacar los troncos de los corrales y llevarlos al area de la leña, despues acomoda a los animales en sus corrales correspondientes.
>
W
A S D
Move

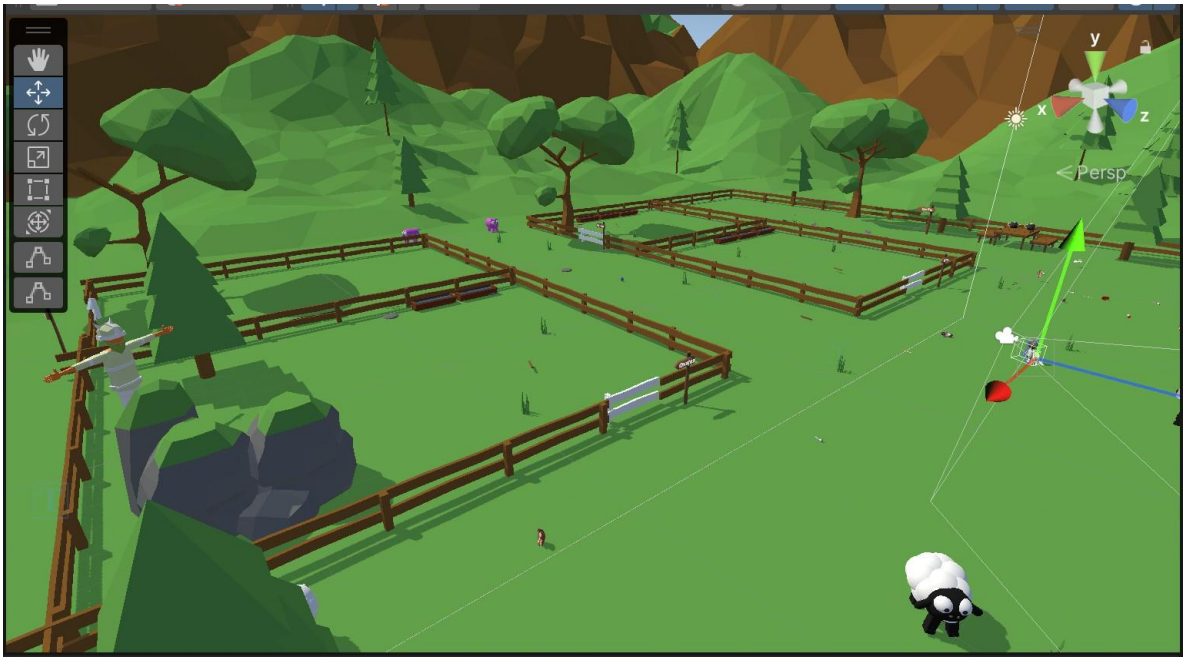

Pulsa la tecla [R] para activar el rayo. Apunta hacia la basura para poder recogerla. ¡Usa su poder para dejar la granja impecable!


Para recoger la leña pulsa la tecla [F] y para lograr que los animales te sigan tocalos.
Una vez finalizadas las tareas acercate con Juan
¡Buena suerte!


¡Hola! Soy Pett, granjero de Chicken Little
Ten un bonito dia


¡Lo lograste! Has realizado todas las tareas!

Una vez adaptado el entorno a como lo queriamos, el diseño quedaria algo tal que asi



En este entorno se ha implementa la posibilidad de mejorar la experiencia de juego haciendo uso de scripts para poder mover la camara para arriba y para los lados, ya sea usando el trackpad de la computadora o con un mouse. Scripts los cuales van en integrados en el jugador.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Mouse_MoverCamara : MonoBehaviour
{
    [SerializeField] Transform camara;

    [SerializeField] float velocidad;
    [SerializeField]
    float anguloX; //el mouse se mueve sobre el eje Y, sin embargo, el objeto (camara) debe rotar sobre su eje X para subir o bajar la mira de la camara

    // Start is called before the first frame update
    void Start()
    {
        anguloX = 0;
    }

    // Update is called once per frame
    void Update()
    {

        float valY = Input.GetAxis("Mouse Y");

        float valY_conVelocidad = valY * velocidad; //* Time.deltaTime //conviene?

        //Debug.Log("Y: [" + valY + " -- "+ valY_conVelocidad+ "]");


        anguloX -= valY_conVelocidad; //* configAxis;//invert axis

        anguloX = Mathf.Clamp(anguloX,-45f, 45f);
        //Debug.Log("Angulo Y: " + anguloX);

        camara.localRotation = Quaternion.Euler(anguloX, 0f, 0f);

    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Mouse_RotarPersonaje : MonoBehaviour
{

    [SerializeField]
    float velocidad;


    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {

        float valX = Input.GetAxis("Mouse X");

        float valX_conVelocidad = valX * velocidad; // * Time.deltaTime; //conviene?

        //Debug.Log("X: [" + valX + " -- "+ valX_conVelocidad+ "]");

        //alternativa -> Vector3.up
        transform.Rotate(new Vector3(0, valX_conVelocidad, 0));

    }
}
```

Para el movimiento del jugador se utilizó el script de movimiento y rotacion compuesto que tambien estuvimos utilizando en clase.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class S4_MovRotCompuesto : MonoBehaviour
{
    [SerializeField] float velocidad_rotacion = 10f;

    [SerializeField] float velocidad_movimiento = 10f;
    // Update is called once per frame
    void Update()
    {

        float angulo = 5f * velocidad_rotacion *  Time.deltaTime;
        //                X   Y   Z
    if(Input.GetKey(KeyCode.Q)){ //giro a la izquierda
        transform.Rotate(0, angulo * -1, 0);
    }else if (Input.GetKey(KeyCode.E)) //giro a la derecha
    {
        transform.Rotate(0, angulo, 0);
}
        //arriba - abajo
        if (Input.GetKey(KeyCode.W)){
            transform.position += transform.forward * velocidad_movimiento * Time.deltaTime;
        }
        else if(Input.GetKey(KeyCode.S)){
            transform.position += transform.forward * -1 * velocidad_movimiento *Time.deltaTime;
        }

        //izquierda - derecha
        if (Input.GetKey(KeyCode.A)){
            transform.position += transform.right * -1 * velocidad_movimiento * Time.deltaTime;
        }
        else if(Input.GetKey(KeyCode.D)){
            transform.position += transform.right *  velocidad_movimiento *Time.deltaTime;
        }
    }
}
```

Por otro lado, todos los animales incluyen algunos scripts que se llegaron a ver en clase como lo es el caso de LookAt, CalcularDistancia y MovimientoEnemigo (que en este caso, no es enemigo, es un animalito).

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CalcularDistancia_Enemigo : MonoBehaviour
{
    Transform  ubi_obj_a_calc_dist;
    float distance;


    public float getDistance(){
        return distance;
    }

    private void Awake(){
        ubi_obj_a_calc_dist = GameObject.Find("Jugador").GetComponent<Transform>();
    }

    // Update is called once per frame
    void Update()
    {
        distance = Vector3.Distance(
            transform.position,
            ubi_obj_a_calc_dist.position);
    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class LookAt_Zombie : MonoBehaviour
{
    Transform ubi_obj_a_mirar;
    CalcularDistancia_Enemigo auxComponenteDistance;

    private void Awake(){
        ubi_obj_a_mirar = GameObject.Find("Jugador").GetComponent<Transform>();
    }
    // Start is called before the first frame update
    void Start()
    {
        auxComponenteDistance = GetComponent<CalcularDistancia_Enemigo>();
    }

    // Update is called once per frame
    void Update()
    {
        float distanciaAlEnemigo = auxComponenteDistance.getDistance();
        if(distanciaAlEnemigo < 3.5f){ // umbral de distancia con el personaje
            float valY = ubi_obj_a_mirar.position.y;

            if (valY>3.0f){// si esta por encima del humbral (3.0f)
            transform.LookAt(new Vector3(
                ubi_obj_a_mirar.position.x,
                3.0f,
                ubi_obj_a_mirar.position.z
            ));
        }
            else{
                transform.LookAt(ubi_obj_a_mirar.position);

            }
    } }  }
//Vinculacion de componentes es asi...
//componentes en el Start
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Movimiento_Enemigo : MonoBehaviour
{
    CalcularDistancia_Enemigo auxComponenteDistance;
    Transform ubi_personaje;

    private void Awake(){
        ubi_personaje = GameObject.Find("Jugador").GetComponent<Transform>();
    }
    // Start is called before the first frame update
    void Start()
    {
        auxComponenteDistance = GetComponent<CalcularDistancia_Enemigo>();

    }

    // Update is called once per frame
    void Update()
    {
        float distanciaAlEnemigo = auxComponenteDistance.getDistance();
        if(distanciaAlEnemigo < 5.5f){ // umbral de distancia con el personaje
            float velocidad = 1f * Time.deltaTime;
            transform.position =
                Vector3.MoveTowards(transform.position, ubi_personaje.position, velocidad);
    }
}
}
```

Para activar el movimiento de los animales (para que no nos comiencen a seguir desde el comienzo, sino hasta que se interactue con ellos) se usó este script, el cual activa el movimiento del animalito solo hasta que el jugador colisione con el animalito, todo esto con ayuda de un trigger.

```
using System.Collections;
using System.Collections.Generic;
using EasyPrimitiveAnimals;
using UnityEngine;

public class ActivarMov_Animals : MonoBehaviour
{

    public bool enemigoMovimientoActivo;
    public bool Animalcontrolleractivo;

    private void Start()
    {

        enemigoMovimientoActivo = false;
        Animalcontrolleractivo = false;


        Movimiento_Enemigo movimiento = GetComponent<Movimiento_Enemigo>();
        if (movimiento != null)
        {
            movimiento.enabled = false;
        }

        AnimalController animalController = GetComponent<AnimalController>();
        if (animalController != null)
        {
            animalController.enabled = false;
        }
    }

    private void OnTriggerEnter(Collider other)
    {

        if (other.CompareTag("Jugador"))
        {
            Movimiento_Enemigo movimiento = GetComponent<Movimiento_Enemigo>();
            if (movimiento != null)
            {
                movimiento.enabled = true;
```

```
30      }
31
32      private void OnTriggerEnter(Collider other)
33      {
34
35          if (other.CompareTag("Jugador"))
36          {
37              Movimiento_Enemigo movimiento = GetComponent<Movimiento_Enemigo>();
38              if (movimiento != null)
39              {
40                  movimiento.enabled = true;
41                  enemigoMovimientoActivo = true;
42                  Debug.Log("Movimiento del enemigo activado.");
43              }
44
45              AnimalController animalController = GetComponent<AnimalController>();
46              if (animalController != null)
47              {
48                  animalController.enabled = true;
49                  Animalcontrolleractivo = true;
50                  Debug.Log("Animal controller activado.");
51              }
52          }
53      }
54  }
55
```

RayoCast

```
using System.Collections;
using System.Collections.Generic;
using TMPro;
using UnityEngine;

public class RayoCast : MonoBehaviour
{
    private int contadorBasura = 0;
    private int limiteBasura = 31;
    private float alcanceRayo = 5f;
    [SerializeField] public GameObject mensajeDinamica2;
    [SerializeField] public GameObject mensajeDinamica2_2;
    [SerializeField] public GameObject contador;


    private Vector3 direccionFija = Vector3.forward;

    private void Start()
    {
        mensajeDinamica2.SetActive(false);
        mensajeDinamica2_2.SetActive(false);
    }

    void Update()
    {
        if (Input.GetKey(KeyCode.R))
        {
            if (contadorBasura >= limiteBasura)
            {
                Debug.Log("Se ha alcanzado el límite de objetos destruidos. Raycast desactivado.");
                contador.SetActive(false);
                mensajeDinamica2.SetActive(true);
                return;
            }

            RaycastHit hit;

            if (Physics.Raycast(transform.position, transform.TransformDirection(direccionFija), out hit, alcanceRayo))
            {
                Debug.Log("Colisiona con un objeto");

                if (hit.collider.CompareTag("Basura"))
                {
                    Debug.Log("Destruyendo objeto: " + hit.collider.gameObject.name);
                    Destroy(hit.collider.gameObject);
                    contadorBasura++;

                    // Actualizar el contador de basura en el UI
                    HandlerUI.Instance.ActualizarContadorBasura(contadorBasura);


                    Debug.Log("Objetos destruidos: " + contadorBasura);
                }
                Debug.DrawRay(transform.position, transform.TransformDirection(direccionFija) * hit.distance, Color.green);
            }
            else
            {
                Debug.Log("No colisiona");
                Debug.DrawRay(transform.position, transform.TransformDirection(direccionFija) * alcanceRayo, Color.red);
            }
        }
    }
    public void cambioMensaje()
    {
        mensajeDinamica2.SetActive(false);
        mensajeDinamica2_2.SetActive(true);
        StartCoroutine(OcultarDialogo(mensajeDinamica2_2));
    }
    private IEnumerator OcultarDialogo(GameObject dialogo)
    {
        yield return new WaitForSeconds(5);
        if (dialogo != null)
        {
            dialogo.SetActive(false);
```

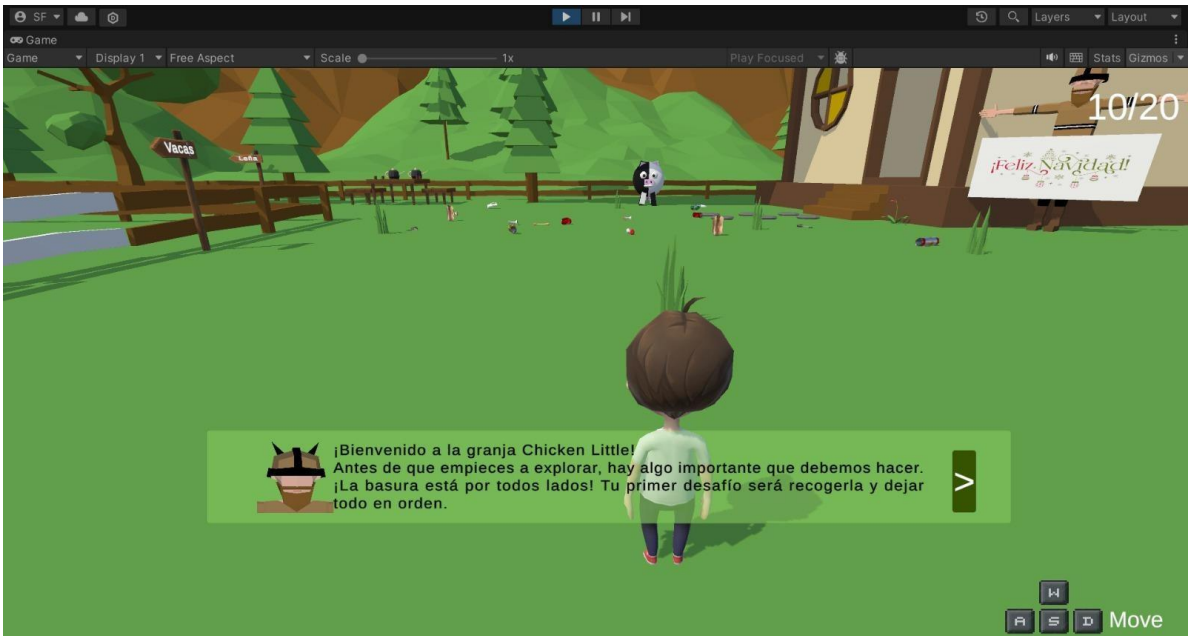
```
            }
        }
    }
    public void cambioMensaje()
    {
        mensajeDinamica2.SetActive(false);
        mensajeDinamica2_2.SetActive(true);
        StartCoroutine(OcultarDialogo(mensajeDinamica2_2));
    }
    private IEnumerator OcultarDialogo(GameObject dialogo)
    {
        yield return new WaitForSeconds(5);
        if (dialogo != null)
        {
            dialogo.SetActive(false);
        }
    }
}

```

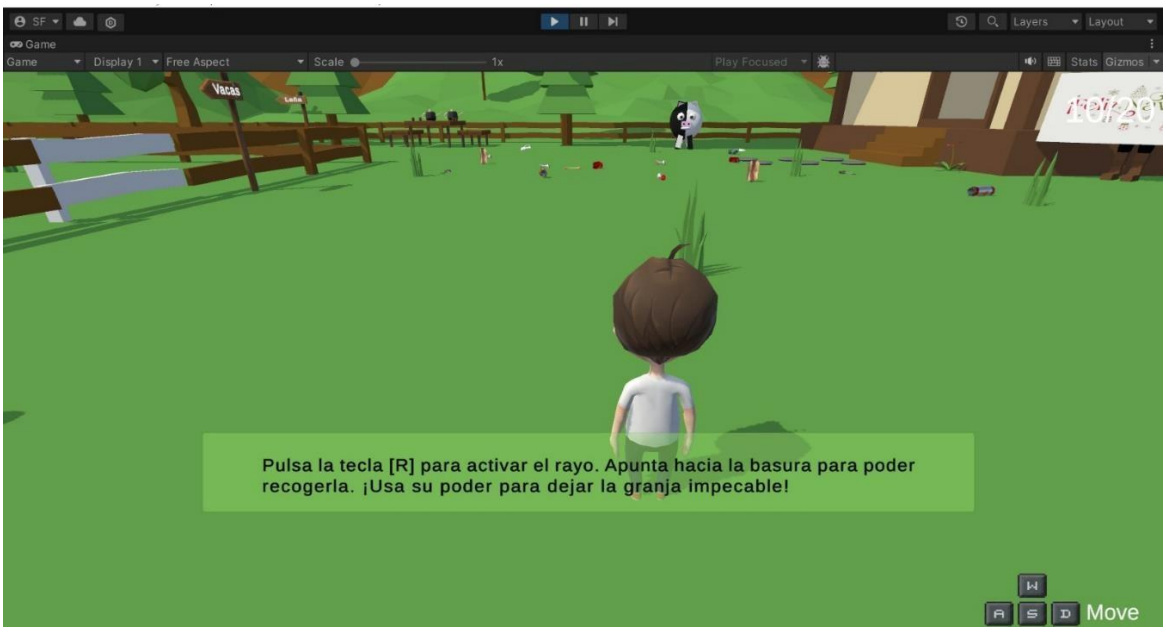
Imágenes de su ejecución puesta en practica.

Al iniciar el juego nos mostrara una pantalla de inicio con un botón para que el juego comience

Al iniciar el juego, nos mostrara en la parte inferior de la pantalla un mensaje que son las indicaciones de las tareas

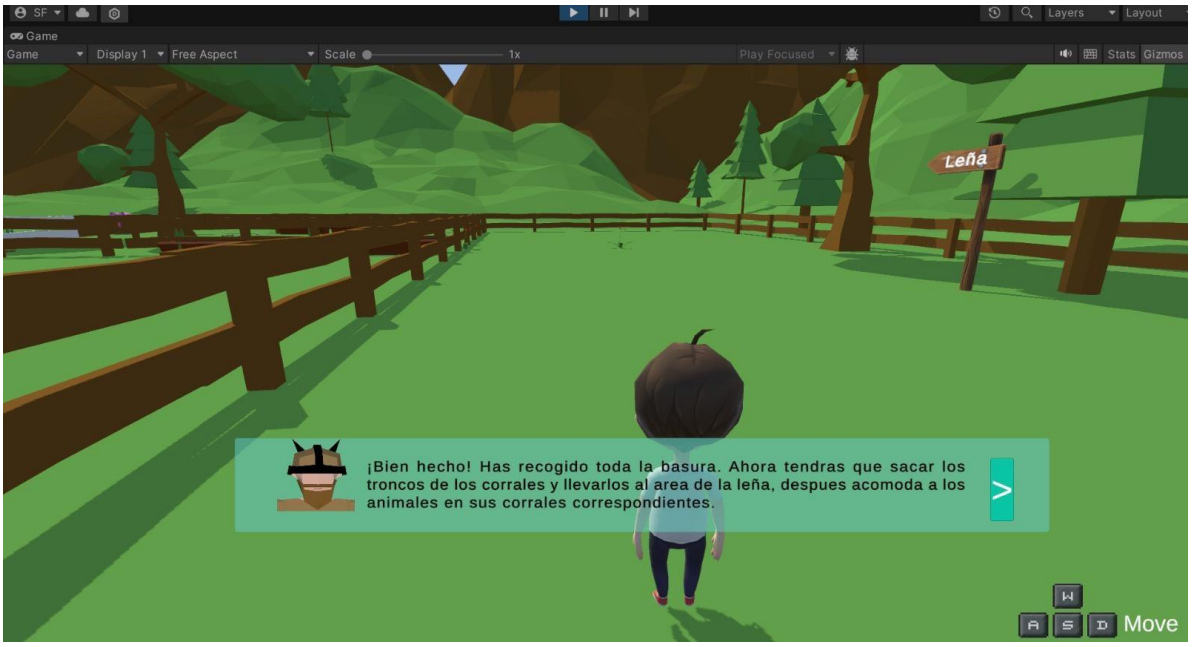


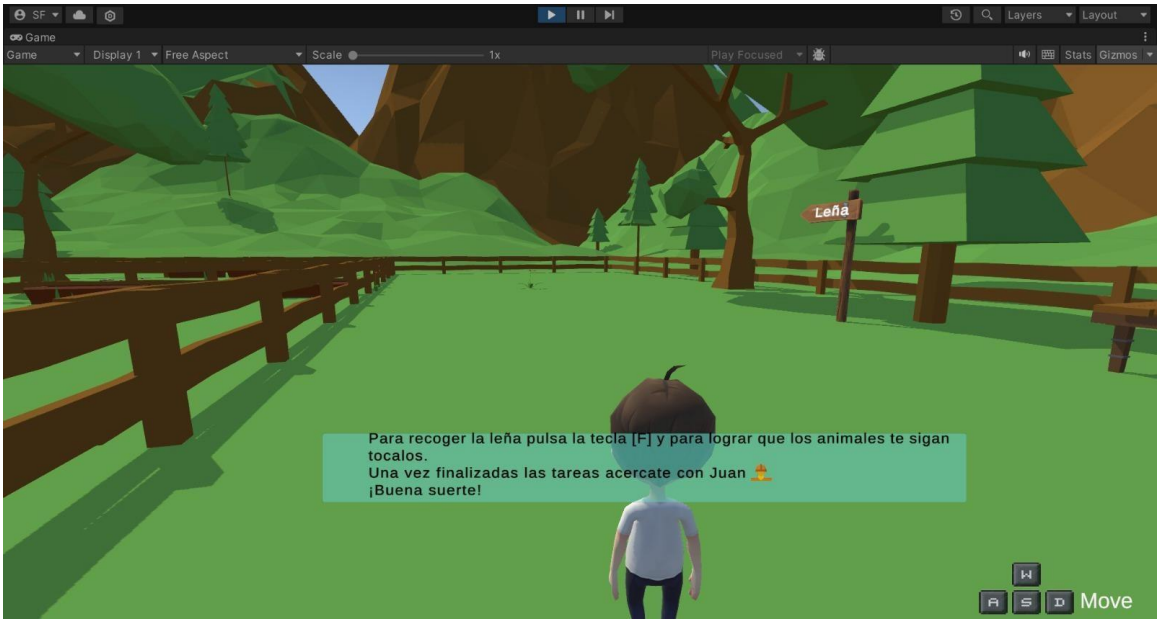
También nos muestra mensajes de con que teclas se puede interactuar



Con el rayo que sale de la cabeza de Timmy podrá eliminar la basura que hay en el mapa

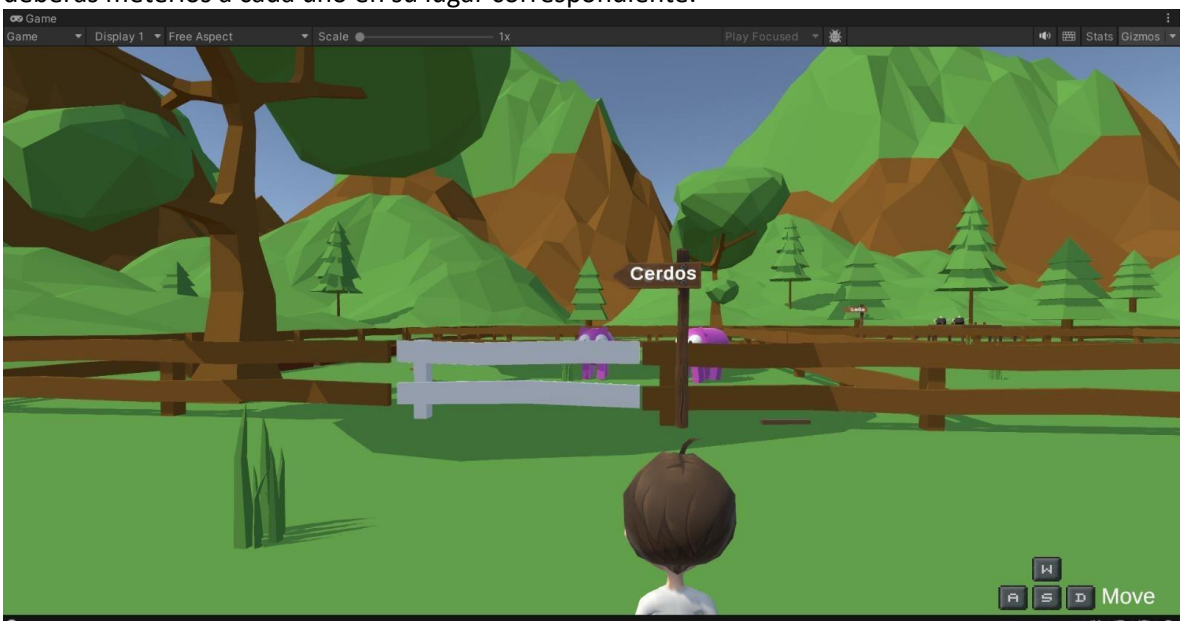
Una vez terminado de eliminar toda la basura, el juego te dara mas indicaciones de las tareas que deberas de realizar.



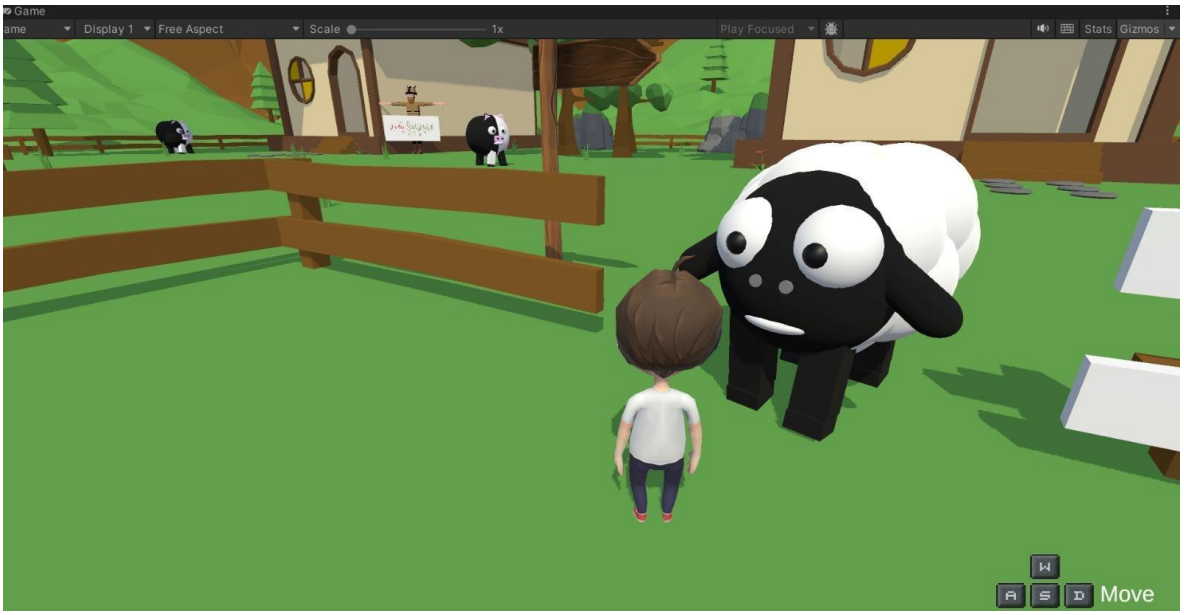


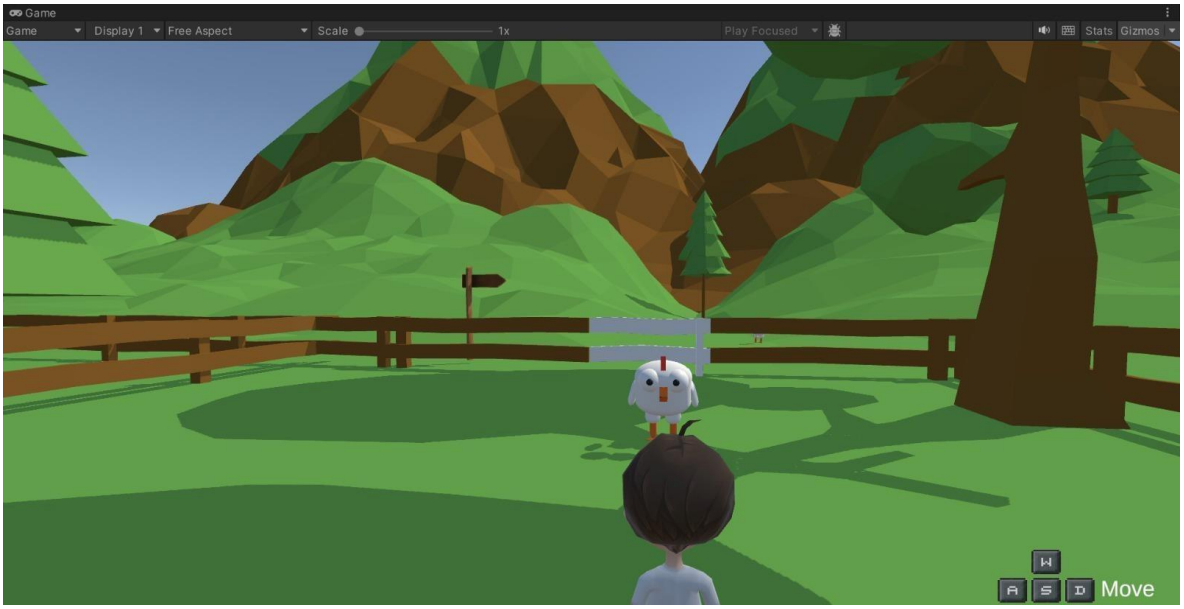
En la siguiente imagen muestra como Timmy puede recoger madera para poder soltarlo en la zona de leña.

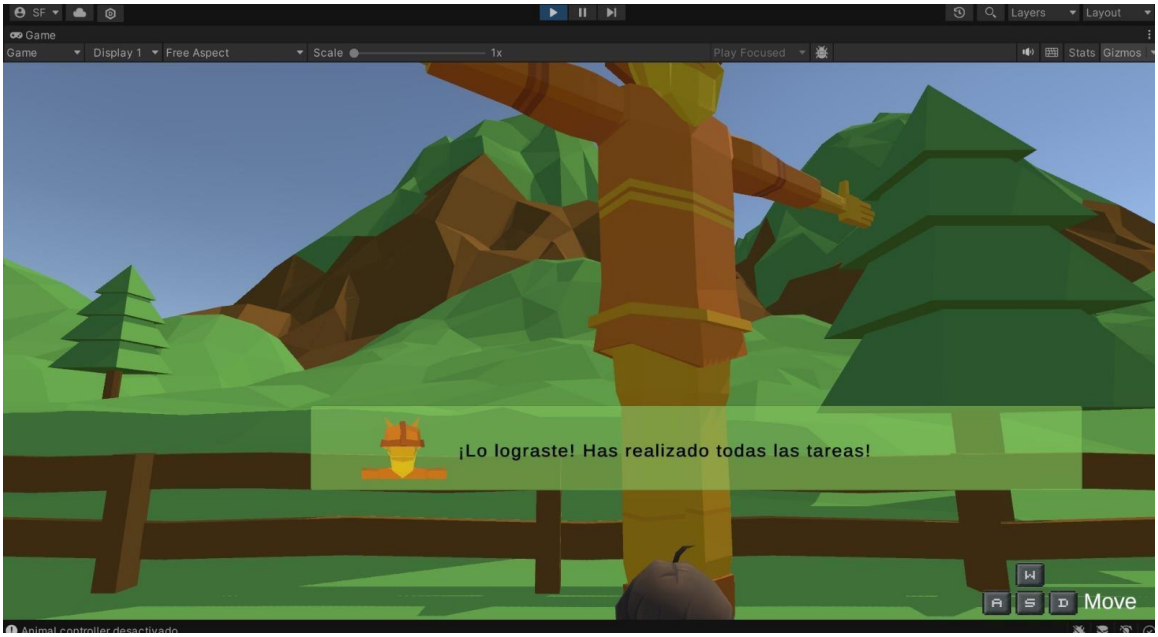
Como siguiente tarea del juego, nos decía en los mensajes que existen corrales para cada animalito, deberás meterlos a cada uno en su lugar correspondiente.



Para esto debes colisionar con el animalito para que este te comience a perseguir y llevarlos hasta su corral.







Una vez terminada las tareas, deberas ir con Juan para finalizar el juego