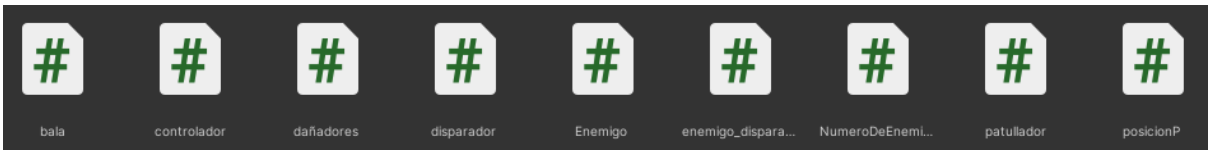
Sistema de IA básica para enemigos en unity 2D

En esta documentación se detallarán los aspectos de la herramienta, así como de los códigos que la componen y los demás apartados para poder manipular las inteligencias.

Apartado interno (códigos)
Para este juegos se utilizaron los siguientes códigos:



Bala.

```
private GameObject player;
private Rigidbody2D rb;
public float force;
private float timer;

// Start is called before the first frame update
void Start()
{
    rb = GetComponent<Rigidbody2D>();
    player = GameObject.FindGameObjectWithTag("Player");

    Vector3 direction = player.transform.position - transform.position;
    rb.velocity = new Vector2(direction.x, direction.y).normalized * force;

    float rot = Mathf.Atan2(-direction.y, -direction.x) * Mathf.Rad2Deg;
    transform.rotation = Quaternion.Euler(0, 0, rot + 90);
}

// Update is called once per frame
void Update()
{
    timer += Time.deltaTime;

    if(timer > 10)
    {
        Destroy(gameObject);
    }
}

void OnTriggerEnter2D(Collider2D other)
{
    if (other.gameObject.CompareTag("Player"))
    {
        Destroy(gameObject);
    }
```

- Projectile que daña al jugador
- Sale cada x cantidad de tiempo
- Una vez que toca al jugador esta se destruye, de no tocar al jugador, la bala se borrará después de un tiempo para no saturar la memoria.

Controlador.

- Controla de forma aleatoria los enemigos que saldrán en la escena
- Saca a los enemigos en un rango delimitado por el usuario, pero es de manera al azar
- El área de aparición depende de cuantos puntos cree el usuario

```
public class controlador : MonoBehaviour
{
    private float minX, maxX, minY, maxY;
    [SerializeField] private Transform[] puntos;
    [SerializeField] private GameObject[] enemigos;
    [SerializeField] private float tiempoEnemigos;
    private float tiempoSiguienteEnemigo;


    // Start is called before the first frame update
    void Start()
    {
        maxX = puntos.Max(puntos => puntos.position.x);
        minX = puntos.Min(puntos => puntos.position.x);
        maxY = puntos.Max(puntos => puntos.position.y);
        minY = puntos.Min(puntos => puntos.position.y);
    }

    // Update is called once per frame
    void Update()
    {
        tiempoSiguienteEnemigo += Time.deltaTime;

        if(tiempoSiguienteEnemigo >= tiempoEnemigos)
        {
            tiempoSiguienteEnemigo = 0;
            CrearEnemigo();
        }
    }

    private void CrearEnemigo()
    {
        int numeroEnemigo = Random.Range(0, enemigos.Length);
        Vector2 posicionAleatoria = new Vector2(Random.Range(minX, maxX), Random.Range(minY, maxY));
```

Dañadores.

```
public class dañadores : MonoBehaviour
{
   [SerializeField] private float tiempoEntreDaño;
   private float tiempoSiguienteDaño;

   internal class CombateJugador
   {
      internal void TomarDaño(int v)
      {
         throw new NotImplementedException();
      }
   }
   private void OnTriggerStay2D(Collider2D other)
   {
      if (other.CompareTag("Player"))
      {
         tiempoSiguienteDaño -= Time.deltaTime;
         if (tiempoSiguienteDaño <= 0)
         {
            other.GetComponent<CombateJugador>().TomarDaño(5);
            tiempoSiguienteDaño = tiempoEntreDaño;
         }
      }
   }
}
```

- Este scrip hace que los enemigos, proyectiles y creaciones de enemigos le hagan daño al jugador
- cada enemigo tiene el suyo y hace una cantidad de daño distinta
- pueden modificarse al gusto

Disparador.

- este scrip es único del enemigo “Tímido”
- genera un sub enemigo el cual se queda estático en la escena
- Este sub enemigo igual puede dañar al jugador

```
public class disparador : MonoBehaviour
{
   public Transform punto_instancai;
   public GameObject bala;
   private float tiempo;

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

   }

   // Update is called once per frame
   void Update()
   {
      tiempo += Time.deltaTime;
      if(tiempo>=10)
      {
         Instantiate(bala, punto_instancai.position, Quaternion.identity);
         tiempo = 0;
      }
   }
}
```

Enemigo.

```
public GameObject bullet;
public Transform bulletpos;

private float timer;
private GameObject player;

// Start is called before the first frame update
void Start()
{
   player = GameObject.FindGameObjectWithTag("Player");
}



// Update is called once per frame
void Update()
{
   float distance = Vector2.Distance(transform.position, player.transform.position);
   Debug.Log(distance);

   if (distance < 20)
   {
      timer += Time.deltaTime;

      if (timer > 2)
      {
         timer = 0;
         shoot();
      }
   }

}

void shoot()
{
   Instantiate(bullet, bulletpos.position, Quaternion.identity);
```

- Este scrip es único del enemigo “Agresivo”
- Es la IA del enemigo
- Genera balas que dañan al jugador
- Si el jugador entra en el rango del enemigo, este empezará a disparar a él

Enemigo-disparador.

- este scrip es único del enemigo “Tímido”
- Le da movimiento al al enemigo, tiene un rango de alcance del jugador
- si está muy cerca del jugador se detendrá
- si el jugador intenta acercarse al enemigo este se alejará de él
- Es la IA del enemigo

```
// Start is called before the first frame update
void Start()
{
    player_pos = GameObject.Find("Player").transform;
}

// Update is called once per frame
void Update()
{
    //movimiento
    #region
    if (Vector2.Distance(transform.position, player_pos.position) > distancia_frenado)
    {
        transform.position = Vector2.MoveTowards(transform.position, player_pos.position, speed * Time.deltaTime);
    }
    if (Vector2.Distance(transform.position, player_pos.position) < distancia_retraso)
    {
        transform.position = Vector2.MoveTowards(transform.position, player_pos.position, -speed * Time.deltaTime);
    }
    if (Vector2.Distance(transform.position, player_pos.position) < distancia_frenado &&
Vector2.Distance(transform.position, player_pos.position) > distancia_retraso)
    {
        transform.position = transform.position;
    }
    #endregion

    //flip
    #region
    if (player_pos.position.x > this.transform.position.x)
    {
        this.transform.localScale = new Vector2(1, 1);
    }
    else
    {
        this.transform.localScale = new Vector2(-1, 1);
    }
```

Número de enemigos.

```
public class NumeroDeEnemigos : MonoBehaviour
{
    public string tagToCount = "NumeroDeEnemigos"; // Etiqueta a contar
    public Text counterText; // Texto donde se mostrar
 el contador


    // Update is called once per frame
    void Update()
    {
        // Obtener la cantidad de objetos con la etiqueta especificada
        int count = GameObject.FindGameObjectsWithTag(tagToCount).Length;

        // Actualizar el texto del contador
        counterText.text = "El numero de objetos generados es: " + count.ToString();
    }
}
```

- Muestra en la consola el número de enemigos que se genera

Patrullador.

- Este scrip es único del enemigo “Patrullador”
- Es la IA del enemigo
- Le da un recorrido que seguir
- Cuando el jugador pasa por la zona en la que el enemigo ronda, este le seguirá
- le hace daño al tocarlo

```
public class patullador : MonoBehaviour
{
    [SerializeField] private float velocidadMovimiento;
    [SerializeField] private Transform[] puntosMovimiento;
    [SerializeField] private float distanciaMinima;

    private int numeroAleatorio;
    private SpriteRenderer sprite;


    // Start is called before the first frame update
    void Start()
    {
        numeroAleatorio = Random.Range(0, puntosMovimiento.Length);
        sprite = GetComponent<SpriteRenderer>();
        Girar();


    }

    // Update is called once per frame
    void Update()
    {
        transform.position = Vector2.MoveTowards(transform.position, puntosMovimiento[numeroAleatorio].position,
velocidadMovimiento * Time.deltaTime);
        if(Vector2.Distance(transform.position, puntosMovimiento[numeroAleatorio].position) < distanciaMinima)
        {
            numeroAleatorio = Random.Range(0, puntosMovimiento.Length);
            Girar();
        }
```

PosicionP.

```
public class posicionP : MonoBehaviour
{
    public TMP_Text posicionText;
    public GameObject targetObject;
    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        posicionText.text = "Posicion del jugador: " + targetObject.transform.position.ToString();
```

- Muestra en pantalla la posición actual del jugador

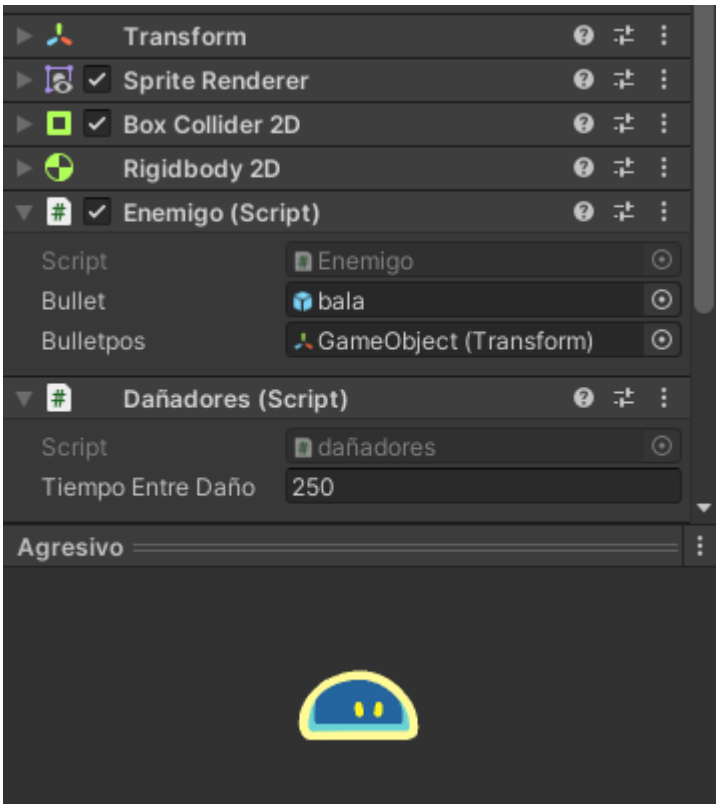
El jugador tiene un único scrip, el cual es:

- Este scrip es único del jugador
- Hace que el jugador se pueda mover por el mapa
- se pueden usar las letras WASD o las flechas del teclado

Apartado visual (juego e inspector)
Enemigos.

Para el juego se realizaron tres enemigos, cada uno con características distintas los cuales son:

Agresivo.



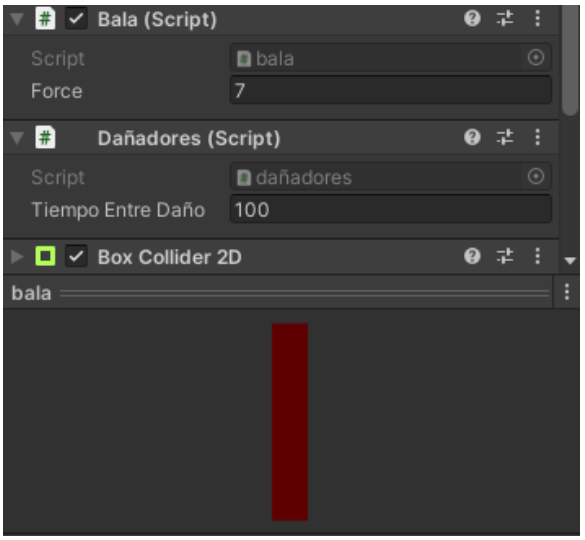
Este enemigo hace:

- seguir al jugador
- disparar proyectiles

Si el jugador entra en su rango de visión, le empezará a disparar

Si se aleja lo suficiente dejará de disparar

La bala del enemigo Agresivo tiene un periodo de vida si es que no le logra dar al jugador.



Patrullador.

Este enemigo hace:

- Rondas de movimiento en los que da vueltas

El enemigo tiene rondas de movimiento, si el jugador llega a tocar una de las rondas, el enemigo empezará a seguirlo.

Si el enemigo toca al jugador, este lo dañará

Tímido.

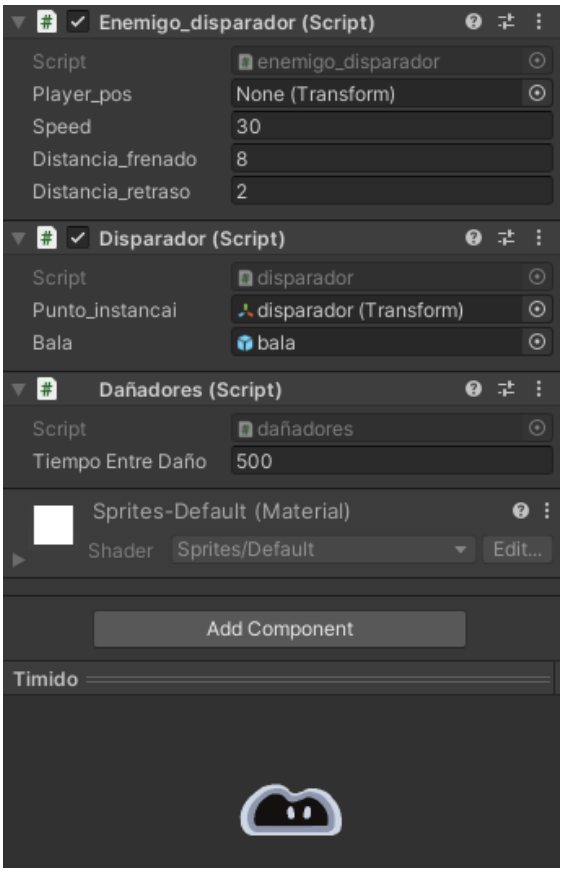
Este enemigo hace:

- Perseguir al jugador
- soltar mini enemigos

Cuando el el jugador se aleja de él, este lo empezará a perseguir

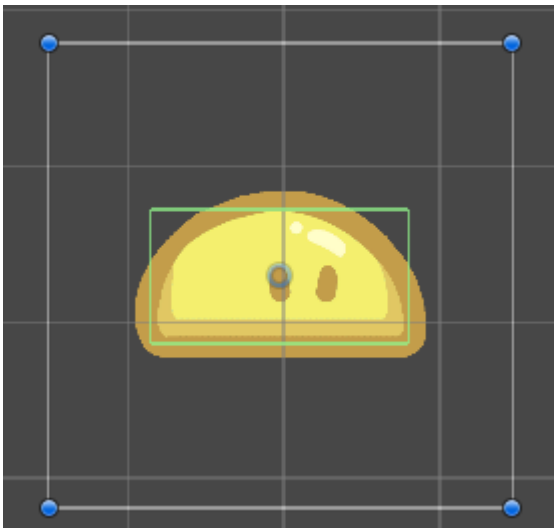
si el jugador se acerca al enemigo, este se alejara de él

Cada cierto tiempo el enemigo soltara pequeños enemigos



Este mini enemigo solo se queda en la posición en la que apareció y daña al jugador.

Jugador.



El jugador se puede mover en todo momento del juego

cuenta con una cámara que lo sigue a todas partes

Tiene un visualizador de su posición en todo momento.

Todos los enemigos destruyen al jugador, si este es destruido no se podrá jugar más, y saldrá una pantalla para reiniciar el juego.

El juego cuenta con un menú de pausa y una pantalla de muerte para que el jugador pueda volver a jugar el juego las veces que quiera.


Posicion del jugador: (-7.02, 0.30, 0.00)
PAUSA
PLAY
Tiempo con vida: 3.495706


Posicion del jugador: (-10.02, 0.30, 0.00)
HAS
PERDIDO
Tiempo con vida: 1.883284


Posicion del jugador: (-7.02, -6.70, 0.00)