El Diver

Proyek Akhir OOP

oleh

Kelompok-22

Anggota Kelompok

Javana Dzaki M

M Arya Wiandra U

Nugroho Ulil A

Rivi Yasha H

STORY LINE

Lore

“*El Diver*: Misteri di Palung Uliana” adalah sebuah game RPG di mana pemain menjadi seorang mantan penyelam profesional yang dijuluki *El Diver*. Saat ini, *El Diver* sedang menikmati masa pensiun nya di sebuah pulau yang dikelilingi oleh Palung Uliana. Suatu hari, nenek *El Diver terkena penyakit* yang sangat langka, El Diver pun memesankan obat untuk neneknya dari negara yang sangat jauh, yang membuat obat ini sangat mahal untuk dibeli.

Sayangnya, ketika helikopter yang bertugas membawakan obat ke pulau tempat tinggal El Diver mengalami kecelakaan. Sekarang, El Diver harus memanfaatkan kemampuan menyelamnya untuk mencari dan memberikan obat untuk neneknya yang sedang sakit.


MECHANICS

Mechanics

Pemain memiliki kendali penuh atas El Diver dan dapat melakukan free-diving, serta panah untuk menangkap dan menembak mengikuti arah kursor

Kontrol Utama:

- ASWD / Arrows untuk berenang
- R untuk mengubah mode antara catchin dan shooting
- Arrow dan gun yang mengikuti arah gerak kursor
- E untuk unequip item
- Diving recap untuk score


INTERFACE

Interface

Player Input:

- ASWD / Arrow keys untuk berenang
- Klik kiri ketika mode tanda panah untuk menangkap ikan
- Klik kanan untuk spawn fish, ubah fish yang akan dispawn dengan input angka 1, 2, 3.

Interaction:

- Ikan pasif untuk ditangkap dan akan dihitung score
- Ikan agresif untuk menambah tingkat kesulitan dan memberi sense survival

Feedback visual:

- + 1 Fish ketika berhasil menangkap ikan
- Indikasi terkena damage dengan blinking merah

Tampilan UI:

- Health bar untuk indikasi hitpoints
- Oxygen bar untuk indikasi sisa napas
- Tampilan diving recap untuk perhitungan skor

ART STYLE

Art Style

Visual:

- Tema pixel art dengan karakter diver, dan memiliki animasi berenang, detail animasi menembak dalam air
- Ikan-ikan yang berenang dan menghindar ketika ingin ditangkap
- Ikan agresif yang menyerang player
- Interactive map yang menggambarkan monumen-monumen.

Audio:

- Ambient music yang membuat kita immersed seperti sedang menyelam ke palung yang dalam untuk meningkatkan rasa adventurous.

In-Game Screenshots

Various fishes ranging from passive to aggressive

In Game Screenshots

In Game Screenshots

Code Screenshots

```
0 references
void Update()
{
    UpdateOxygen();
}

2 references
public void UpdateOxygen()
{
    currentOxygen -= oxygenDepletionRate * Time.deltaTime;
    if (currentOxygen < 0)
    {
        currentOxygen = 0;
    }
    oxygenBar.SetOxygen(currentOxygen);
}
```

```
0 references
void Start()
{
    if (oxygenBar == null)
    {
        GameObject oxygenBarObject = GameObject.Find("OxygenBar");
        if (oxygenBarObject != null)
        {
            oxygenBar = oxygenBarObject.GetComponent<OxygenBar>();
        }
    }

    currentOxygen = maxOxygen;
    oxygenBar.SetMaxOxygen(maxOxygen);
}
```

Sistem oksigen, yang akan terus menerus menurun dan akan tidak sadar jika oksigen habis

Code Screenshots

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

[RequireComponent(typeof(BoxCollider2D))]
2 references
public class AttackComponent : MonoBehaviour
{
    0 references
    public GunBullet bullet;
    2 references
    public int damage;

    0 references
    void Start()
    {
        GetComponent<BoxCollider2D>().isTrigger = true;
    }

    0 references
    void OnTriggerEnter2D(Collider2D other)
    {
        Transform highestParent = GetHighestParent(transform);
        Transform otherHighestParent = GetHighestParent(other.transform);

        if ((highestParent.CompareTag("FishFriendly") && otherHighestParent.CompareTag("FishFriendly")) ||
            (highestParent.CompareTag("FishFriendly") && otherHighestParent.CompareTag("FishEnemy")) ||
            (highestParent.CompareTag("FishEnemy") && otherHighestParent.CompareTag("FishFriendly")) ||
            (highestParent.CompareTag("FishEnemy") && otherHighestParent.CompareTag("FishEnemy")))
        {
            return;
        }

        DealDamage(other);
    }
```

```
1 reference
private void DealDamage(Collider2D other)
{
    InvincibilityComponent invincibility = other.GetComponent<InvincibilityComponent>();
    if (invincibility != null)
    {
        invincibility.StartInvincibility();

        HitboxComponent hitbox = other.GetComponent<HitboxComponent>();
        if (hitbox != null)
        {
            GunBullet bullet = GetComponent<GunBullet>();
            if (bullet != null)
            {
                hitbox.Damage(bullet);
            }
            else
            {
                hitbox.Damage(damage);
            }
        }
    }
}
```

Attacking mechanism untuk menyerang ikan aggresif

Code Screenshots

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

[RequireComponent(typeof(BoxCollider2D))]
3 references
public class HitboxComponent : MonoBehaviour
{
    3 references
    public HealthComponent health;

    0 references
    void Start()
    {
        health = GetComponent<HealthComponent>();
    }

    1 reference
    public void Damage(int damage)
    {
        health.Subtract(damage);
    }

    1 reference
    public void Damage(GunBullet bullet)
    {
        AttackComponent attackComponent = bullet.GetComponent<AttackComponent>();
        if (attackComponent != null)
        {
            health.Subtract(attackComponent.damage);
        }
    }
}
```

Menggunakan Hitbox component agar bisa menerima demage dari ikan aggresif

Code Screenshots

```
public class GameManager : MonoBehaviour
{
    9 references
    public static GameManager Instance { get; private set; }
    8 references
    public LevelManager LevelManager { get; private set; }

    9 references
    private GameObject mainCamera;
    7 references
    private GameObject player;

    0 references
    private void Awake()
    {
        if (Instance == null)
        {
            Instance = this;
            DontDestroyOnLoad(gameObject);

            LevelManager = GetComponentInChildren<LevelManager>();

            mainCamera = GetComponentInChildren<Camera>().gameObject; // Ensure this is the GameManager's camera
            player = GameObject.FindWithTag("Player");
            if (mainCamera != null)
            {
                DontDestroyOnLoad(mainCamera);
            }

            if (player != null)
            {
                DontDestroyOnLoad(player);
            }

            SceneManager.sceneLoaded += OnSceneLoaded;
        }
        else
```

```
public class GameManager : MonoBehaviour
    1 reference
    private void OnSceneLoaded(Scene scene, LoadSceneMode mode)
    {
        if (player == null)
        {
            player = GameObject.FindWithTag("Player");
            if (player != null)
            {
                DontDestroyOnLoad(player);
            }
        }

        GameObject sceneCamera = GameObject.FindWithTag("MainCamera");

        if (scene.name == "level1" || scene.name == "level2" || scene.name == "level3")
        {
            if (sceneCamera != null && sceneCamera != mainCamera)
            {
                sceneCamera.SetActive(false);
            }
            if (mainCamera != null)
            {
                mainCamera.SetActive(true);
                Debug.Log("adasad");
            }
        }
        else
        {
            if (sceneCamera != null && sceneCamera != mainCamera)
            {
                sceneCamera.SetActive(true);
            }
            if (mainCamera != null)
            {
                mainCamera.SetActive(false);
            }
```

Melakukan pengaturan transisi antar level, ditentukan juga bagian mana yang Destroy dan DontDestroyOnLoad

Code Screenshots

```
using UnityEngine;

0 references
public class ChangeLevel : MonoBehaviour
{
    1 reference
    [SerializeField] string sceneName; // Tambahkan variabel untuk menyimpan nama scene

    0 references
    void Start()
    {
        // Initialization code here
    }

    0 references
    void Update()
    {
        // Update code here
    }

    0 references
    private void OnTriggerEnter2D(Collider2D other)
    {
        if (other.CompareTag("Player"))
        {
            GameManager.Instance.LevelManager.LoadScene(sceneName); // Panggil metode Load
        }
    }
}
```

Menentukan level yang dituju berdasarkan input yang diterima di serialize field

Code Screenshots

```
using UnityEngine;

0 references
public class ChangeLevel : MonoBehaviour
{
    1 reference
    [SerializeField] string sceneName; // Tambahkan variabel untuk menyimpan nama scene

    0 references
    void Start()
    {
        // Initialization code here
    }

    0 references
    void Update()
    {
        // Update code here
    }

    0 references
    private void OnTriggerEnter2D(Collider2D other)
    {
        if (other.CompareTag("Player"))
        {
            GameManager.Instance.LevelManager.LoadScene(sceneName); // Panggil metode Load
        }
    }
}
```

Menentukan level yang akan dituju berdasarkan input yang diterima di serialize field

Code Screenshots

```
using UnityEngine;
using UnityEngine.SceneManagement;

0 references
public class DeadScene : MonoBehaviour
{
    0 references
    void Update()
    {
        if (Input.anyKeyDown)
        {
            GameManager.Instance.LevelManager.LoadScene("land");
        }
    }
}
```

Menjalankan Dead Scene jika kehabisan napas atau hitpoints

Code Screenshots

```
public class ReturnShipManager : MonoBehaviour
{
    6 references
    public Image uiImage;
    3 references
    private bool playerInArea = false;

    0 references
    void Start()
    {
        if (uiImage != null)
        {
            uiImage.enabled = false;
        }
    }

    0 references
    void OnTriggerEnter2D(Collider2D other)
    {
        if (other.CompareTag("Player"))
        {
            playerInArea = true;
            if (uiImage != null)
            {
                uiImage.enabled = true;
            }
        }
    }
```

```
public class ReturnShipManager : MonoBehaviour
    0 references
    void OnTriggerExit2D(Collider2D other)
    {
        if (other.CompareTag("Player"))
        {
            playerInArea = false;
            if (uiImage != null)
            {
                uiImage.enabled = false;
            }
        }
    }

    0 references
    void Update()
    {
        if (playerInArea && Input.GetKeyDown(KeyCode.H))
        {
            SceneManager.LoadScene("Recap"); // Load the "Recap"
        }
    }
}
```

Mengatur apabila kita kembali ke layar apa saja yang berubah

Code Screenshots

```
public class menuController : MonoBehaviour
{

    7 references
    public int index;
    4 references
    [SerializeField] bool keyDown;
    2 references
    [SerializeField] int maxIndex;

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

    // Update is called once per frame
    0 references
    void Update()
    {
        if (Input.GetAxis("Vertical") != 0)
        {
            if (!keyDown)
            {
                if (Input.GetAxis("Vertical") < 0)
                {
                    if (index < maxIndex)
                    {
                        index++;
                    }
                    else
                    {
                        index = 0;
                    }
                }
                else if (Input.GetAxis("Vertical") > 0)
```

```
public class menuController : MonoBehaviour
    void Update()
                    else
                    {
                        index = 0;
                    }
                }
                else if (Input.GetAxis("Vertical") > 0)
                {
                    if (index > 0)
                    {
                        index--;
                    }
                    else
                    {
                        index = maxIndex;
                    }
                }
                keyDown = true;
            }
        }
        else
        {
            keyDown = false;
        }
    }
    1 reference
    public void PlayGame()
    {
        GameManager.Instance.levelManager.LoadScene("Land");
    }

    1 reference
    public void QuitGame()
    {
        Application.Quit();
        Debug.Log("Quit!");
```

mengatur UI dari Main Menu

Code Screenshots

```
public class menuButton : MonoBehaviour
{
    3 references
    [SerializeField] menuController menuControllerReference;
    5 references
    [SerializeField] Animator animator;

    2 references
    [SerializeField] int thisIndex;

    2 references
    private bool start = true;

    0 references
    void Start()
    {
        if (thisIndex == 1) { start = false; }
    }
```



```
public class menuButton : MonoBehaviour
{
    0 references
    void Update()
    {
        if (menuControllerReference.index == thisIndex)
        {
            animator.SetBool("selected", true);
            if (Input.GetAxis("Submit") == 1)
            {
                animator.SetBool("pressed", true);
                if (start)
                {
                    menuControllerReference.PlayGame();
                }
                else
                {
                    menuControllerReference.QuitGame();
                }
            }
            else if (animator.GetBool("pressed"))
            {
                animator.SetBool("pressed", false);
            }
        }
        else
        {
            animator.SetBool("selected", false);
        }
    }
}
```

Mengatur button akan menavigasi ke berbagai scene

Code Screenshots

```
public class Player : MonoBehaviour
    [SerializeField] public bool boughtGun;

    void Start()
    {
        playerMovement = GetComponent<PlayerMovement>();
        animator = transform.Find("PlayerVisual").GetComponent<Animator>();
        healthComponent = GetComponent<HealthComponent>();
        oxygenComponent = GetComponent<OxygenComponent>();
    }

    void Update()
    {
        if (oxygenComponent != null || healthComponent != null)
        {
            oxygenComponent.UpdateOxygen();
            healthComponent.UpdateHealth();
            if (healthComponent.currentHealth <= 0 || oxygenComponent.currentOxygen <= 0)
            {
                Die();
            }
        }
    }

    void LateUpdate()
    {
        bool isSwimming = playerMovement.IsSwimming();
        animator.SetBool("isSwimming", isSwimming);
        if (isSwimming)
        {
            animator.Play("PlayerSwim");
        }
    }

    void OnTriggerEnter2D(Collider2D other)
    {
        if (other.CompareTag("FishEnemy"))
        {
            Debug.Log("Player collided with enemy fish.");
            EnemyFishMovement enemyFish = other.GetComponent<EnemyFishMovement>();
            if (enemyFish != null)
```

Kode untuk mengatur status hitpoints player, oxygen level, movement dan on trigger apabila mengenai ikan

```
        return fishCount;
    }

    1 reference
    public void SetFishCount(int count)
    {
        fishCount = count;
    }

    2 references
    public int GetMoneyCount()
    {
        return moneyCount;
    }

    0 references
    public void SetMoneyCount(int count)
    {
        moneyCount = count;
    }

    0 references
    public void IncrementFishCount()
    {
        fishCount++;
    }

    1 reference
    private void Die()
    {
        fishCount = 0;
        GameManager.Instance.LevelManager.LoadScene("Dead");
```

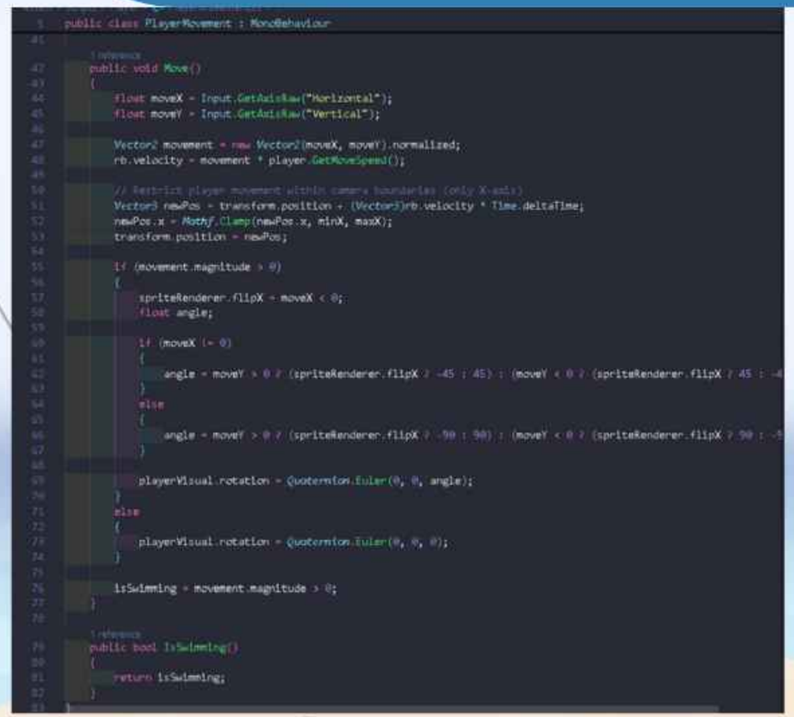
Kode untuk get dan set variabel fish count dan lain lain serta method apabila player mati

Code Screenshots

```
public class PlayerCamera : MonoBehaviour
{
    2 references | 2 references | 2 references | 2 references
    private float minX, maxX, minY, maxY;
    4 references | 3 references
    private float camHalfHeight, camHalfWidth;

    0 references
    void Start()
    {
        target = GameObject.FindGameObjectWithTag("Player").transform;
        GameObject canvasObject = GameObject.Find("Canvas");
        if (canvasObject != null)
        {
            canvasRect = canvasObject.GetComponent<RectTransform>();
            if (canvasRect != null)
            {
                Vector3[] canvasCorners = new Vector3[4];
                canvasRect.GetWorldCorners(canvasCorners);
                minX = canvasCorners[0].x;
                maxX = canvasCorners[2].x;
                minY = canvasCorners[0].y;
                maxY = canvasCorners[2].y;

                camHalfHeight = Camera.main.orthographicSize;
                camHalfWidth = camHalfHeight * Camera.main.aspect;
            }
        }
    }

    0 references
    void Update()
    {
        if (target != null)
        {
            Vector3 newPos = new Vector3(target.position.x, target.position.y + yOffset, -10f);

            // Batasi posisi kamera berdasarkan batas-batas canvas dan ukuran kamera
            newPos.x = Mathf.Clamp(newPos.x, minX + camHalfWidth, maxX - camHalfWidth);
            newPos.y = Mathf.Clamp(newPos.y, minY + camHalfHeight, maxY - camHalfHeight);

            transform.position = Vector3.Slerp(transform.position, newPos, FollowSpeed * Time.deltaTime);
        }
    }
```

Kode untuk mengikuti player dengan camera sehingga player selalu terlihat jelas dan gameplay enak.

Code Screenshots



```
public class PlayerMovement : MonoBehaviour

    1 reference
    public void Move()
    {
        float moveX = Input.GetAxisRaw("Horizontal");
        float moveY = Input.GetAxisRaw("Vertical");

        Vector2 movement = new Vector2(moveX, moveY).normalized;
        rb.velocity = movement * player.GetMoveSpeed();

        // Restrict player movement within camera boundaries (only X-axis)
        Vector3 newPos = transform.position + (Vector3)rb.velocity * Time.deltaTime;
        newPos.x = Mathf.Clamp(newPos.x, minX, maxX);
        transform.position = newPos;

        if (movement.magnitude > 0)
        {
            spriteRenderer.flipX = moveX < 0;
            float angle;

            if (moveX != 0)
            {
                angle = moveY > 0 ? (spriteRenderer.flipX ? -45 : 45) : (moveY < 0 ? (spriteRenderer.flipX ? 45 : -4
            }
            else
            {
                angle = moveY > 0 ? (spriteRenderer.flipX ? -90 : 90) : (moveY < 0 ? (spriteRenderer.flipX ? 90 : -9
            }

            playerVisual.rotation = Quaternion.Euler(0, 0, angle);
        }
        else
        {
            playerVisual.rotation = Quaternion.Euler(0, 0, 0);
        }

        isSwimming = movement.magnitude > 0;
    }

    1 reference
    public bool IsSwimming()
    {
        return isSwimming;
    }
}
```

Kode detail tentang movement player dan bagaimana boolean digunakan untuk mengupdate animasi dari player.

Code Screenshots

```
public class GunBullet : MonoBehaviour
{
    [Header("Bullet Stats")]
    1 reference
    public float bulletSpeed = 35f;
    3 references
    private Rigidbody2D rb;
    1 reference
    private float maxAge = 2f;
    1 reference
    private float damage;

    0 references
    void Awake()
    {
        rb = GetComponent<Rigidbody2D>();
        if (rb != null)
        {
            rb.velocity = transform.up * bulletSpeed;
        }
        Destroy(gameObject, maxAge);
    }

    1 reference
    public void SetDamage(float damage)
    {
        this.damage = damage;
    }

    0 references
    void OnTriggerEnter2D(Collider2D other)
    {
        Destroy(gameObject);
    }
}
```

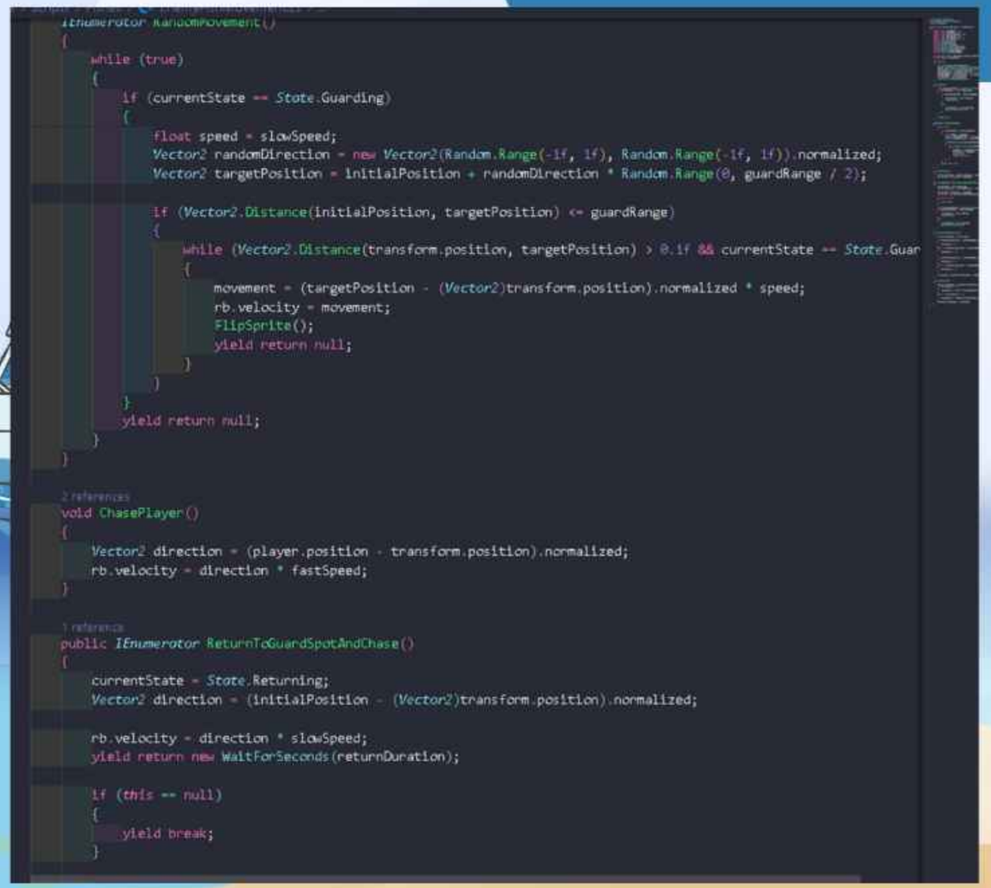
Menggunakan Senjata untuk player

Code Screenshots

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

0 references
public class FishClickSpawner : MonoBehaviour
{
    4 references
    [SerializeField] private GameObject[] fishVariants;
    3 references
    [SerializeField] private int selectedVariant = 0;

    0 references
    void Start()
    {
        Assert.IsTrue(fishVariants.Length > 0, "Please add at least one Fish Prefab!");
    }

    0 references
    void Update()
    {
        for (int i = 1; i <= fishVariants.Length; i++)
        {
            if (Input.GetKeyDown(i.ToString()))
            {
                selectedVariant = i - 1;
            }
        }

        if (Input.GetMouseButtonDown(1))
        {
            SpawnFish();
        }
    }

    1 reference
    private void SpawnFish()
    {
        if (selectedVariant < fishVariants.Length)
        {
            Vector3 spawnPosition = Camera.main.ScreenToWorldPoint(Input.mousePosition);
            spawnPosition.z = 0;
            Instantiate(fishVariants[selectedVariant], spawnPosition, Quaternion.identity);
        }
    }
}
```

Fish spawner
dengan cara click

Code Screenshots



```
IEnumerator RandomMovement()
{
    while (true)
    {
        if (currentState == State.Guarding)
        {
            float speed = slowSpeed;
            Vector2 randomDirection = new Vector2(Random.Range(-1f, 1f), Random.Range(-1f, 1f)).normalized;
            Vector2 targetPosition = initialPosition + randomDirection * Random.Range(0, guardRange / 2);

            if (Vector2.Distance(initialPosition, targetPosition) <= guardRange)
            {
                while (Vector2.Distance(transform.position, targetPosition) > 0.1f && currentState == State.Guar
                {
                    movement = (targetPosition - (Vector2)transform.position).normalized * speed;
                    rb.velocity = movement;
                    FlipSprite();
                    yield return null;
                }
            }
        }
        yield return null;
    }
}

2 references
void ChasePlayer()
{
    Vector2 direction = (player.position - transform.position).normalized;
    rb.velocity = direction * fastSpeed;
}

1 reference
public IEnumerator ReturnToGuardSpotAndChase()
{
    currentState = State.Returning;
    Vector2 direction = (initialPosition - (Vector2)transform.position).normalized;

    rb.velocity = direction * slowSpeed;
    yield return new WaitForSeconds(returnDuration);

    if (this == null)
    {
        yield break;
    }
```

Movement enemy fish dan bagaimana cara enemy fish mengikuti player

Code Screenshots

```
0 references
void Start()
{
    rb = GetComponent<Rigidbody2D>();
    player = GameObject.FindGameObjectWithTag("Player").transform;
    boxCollider = GetComponent<BoxCollider2D>();
    StartCoroutine(RandomMovement());
}

0 references
void Update()
{
    float distanceToPlayer = Vector2.Distance(transform.position, player.position);
    if (distanceToPlayer < detectionRange)
    {
        isFleeing = true;
        FleeFromPlayer();
    }
    else
    {
        isFleeing = false;
    }

    FlipSprite();
}

2 references
IEnumerator RandomMovement()
{
    while (true)
    {
        if (!isFleeing)
        {
            float speed = Random.Range(slowSpeed, fastSpeed);
            movement = new Vector2(Random.Range(-1f, 1f), Random.Range(-0.5f, 0.5f)).normalized * speed;
            rb.velocity = movement;
        }
        yield return new WaitForSeconds(Random.Range(1f, 3f));
    }
}

1 reference
void FleeFromPlayer()
{
```

Movement friendly fish dan bagaimana cara friendly fish behaviour bisa kabur dari player

Code Screenshots

```
public class PlayerFisher : PlayerItem
{
    1 reference
    public FishCounter fishCounter;
    1 reference
    [SerializeField] private Material FisherDefaultMaterial;
    1 reference
    [SerializeField] private Material FisherGreenMaterial;
    2 references
    [SerializeField] private GameObject scorePopupPrefab;
    3 references
    private bool isCollidingWithFishFriendly = false;
    6 references
    private GameObject collidingFishFriendly;
    1 reference
    private Player player;

    1 reference
    protected override void Start()
    {
        spriteRenderer = GetComponent<SpriteRenderer>();
        spriteRenderer.enabled = true;
        boxCollider = GetComponent<BoxCollider2D>();
        boxCollider.enabled = true;
        player = GetComponentInParent<Player>();
    }
```

```
public class PlayerFisher : PlayerItem
    0 references
    void Update()
    {
        if (isEnabled)
        {
            RotateTowardsMouse();
        }

        if (isCollidingWithFishFriendly && Input.GetMouseButtonDown(0))
        {
            UseItem();
        }
    }

    2 references
    public override void ToggleItem()
    {
        isEnabled = !isEnabled;
        spriteRenderer.enabled = isEnabled;
        boxCollider.enabled = isEnabled;
    }

    2 references
    public override void UseItem()
    {
        if (collidingFishFriendly != null)
        {
            collidingFishFriendly.GetComponent<FriendlyFishMovement>().ShrinkAndMoveTowardsPlayer();
            ShowScorePopup();
            fishCounter.AddFish(1);
        }
    }
```

Kemampuan player untuk menggunakan mengambil serta menyimpanikan

Link Video

https://youtu.be/PUjl2a45RJA

TERIMA KASIH

(DM)

Kelompok 22