DAFTAR PUSTAKA

Azzam, M., Habiburrahman, A.-G., Aminul Akbar, M., & Afirianto, T. (2021). *Pengembangan Kecerdasan Buatan Objek NPC pada Micro-Games: Karting menggunakan Finite State Machine Unity* (Vol. 5, Issue 11). http://j-ptiik.ub.ac.id

Ángel Carsí, J., Montero Reyno, E., & Á Carsí Cubel, J. (2008). *Model Driven Game Development: 2D Platform Game Prototyping. MULTIPLE: Multimodeling Approach for Quality-Aware Software Product Lines View project PRISMA: Aspect-Oriented Software Architectures View project MODEL-DRIVEN GAME DEVELOPMENT: 2D PLATFORM GAME PROTOTYPING*. https://www.researchgate.net/publication/221024315

Gamedevbeginner.com. "*Raycasts in Unity, made easy*". Tersedia di : https://gamedevbeginner.com/raycasts-in-unity-made-easy/ diakses tanggal 5 Januari 2022.

Ningrum, F. C., Suherman, D., Aryanti, S., Prasetya, H. A., & Saifudin, A. (2019). Pengujian Black Box pada Aplikasi Sistem Seleksi Sales Terbaik Menggunakan Teknik Equivalence Partitions. *Jurnal Informatika Universitas Pamulang*, *4*(4), 125-130.

Prasetyo, Eko Teguh, Nurcahyani Dewi Retnowati. T.(2015). *Penerapan Kecerdasan Buatan Pada Game "Air Strike Stta" Berbasis Client Server Pada Android*. (Vol. 4, Issue 1). http://dx.doi.org/10.28989/compiler.v4i1.83

Rahmawati, I., Leksono, I. P., & Harwanto, H. (2020). Pengembangan Game Petualang untuk Pembelajaran Berhitung. *Edcomtech: Jurnal Kajian Teknologi Pendidikan*, *5*(1), 11-23.

Suryadi, A. (2018). *Perancangan Aplikasi Game Edukasi Menggunakan Model Waterfall*. *Jurnal Petik*, *3*(1), 8. https://doi.org/10.31980/jpetik.v3i1.352

Sibero, Ivan C. 2009. *"Langkah Mudah Membuat Game 3D."* Yogyakarta: Penerbit MediaKom.

Sagala, M. L., Adams Jonemaro, E. M., & Wardhono, W. S. (2017). *Pengembangan Game Platformer 2D Menggunakan Teknik Projection Mapping* (Vol. 1, Issue 11). http://j-ptiik.ub.ac.id

Teguh Martono, Kurniawan.(2015). *Pengembangan Game Dengan Menggunakan Game Engine Game Maker. Jurnal Sistem Komputer* (Vol. 5, Issue 1).

Tridhonanto, Al. Agency Beranda, 2011. "Optimalkan Potensi Anak dengan Game." Jakarta : Penerbit PT Elex Media Komputindo.

Tristianto, Chrisantus. "Penggunaan metode waterfall untuk pengembangan sistem monitoring dan evaluasi pembangunan pedesaan." *Jurnal ESIT (E-Bisnis, Sistem Informasi, Teknologi Informasi)* 12.1 (2022).

Unity. (2022[a], Agustus 30). Scripting API : Scene. Retrieved Februari 16, 2021 from Unity Documentation: https://docs.unity3d.com/ScriptReference/SceneManagement.Scene.html

Unity. (2022[b] , Agustus 30). Scripting API : Game Object. Retrieved Februari 16, 2022,

from Unity Documentation: https://docs.unity3d.com/ScriptReference/GameObject.html

Unity. (2022[c] , Agustus 30). Unity Scripting API : Component. Retrieved Februari 16, 2022,

from Unity Documentation: https://docs.unity3d.com/ScriptReference/Component.html

Unity. (2022[c] , Agustus 30). Unity Scripting API : Component. Retrieved Maret 5, 2022,

from Unity Documentation: https://docs.unity3d.com/ScriptReference/Component.html

LAMPIRAN

1. Kode penggunaan raycast pada Enemy

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

public class VampireController : MonoBehaviour
{
    [SerializeField]
    Transform player;

    [SerializeField]
    Transform castPoint;

    [SerializeField]
    float agroRange;

    [SerializeField]
    public float ms;

    public GameObject bulletEnemy;
    public Transform shotPointEnemy;

    private float currentTimeSpawn;
    public float timeToSpawn;

    public bool isFacingright;

    bool isAgro = false;
    bool isSearching = false;



    public Transform healthBarEnemy;


    Vector2 firstPos;
    Rigidbody2D rb2d;

    public void Awake()
    {
        rb2d = GetComponent<Rigidbody2D>();
        firstPos = transform.position;
    }
    void Update()
    {
```

```
        if (CanSeePlayer(agroRange))
        {
            isAgro = true;
        }
        else
        {
            if (isAgro)
            {
                if (!isSearching)
                {
                    isSearching = true;

                    Invoke("stopChasePlayer", 2);
                }
            }

        }

        if (isAgro)
        {
            ChasePlayer();
        }
        else
        {
            Patrol();
        }

        bool CanSeePlayer(float distance)
        {
            bool val = false;
            float castDist = distance;

            if (isFacingright)
            {
                castDist = -distance;
            }
            else
            {
                castDist = distance;
            }

            Vector2 endPos = castPoint.position + Vector3.left * castDist;

            RaycastHit2D hit = Physics2D.Linecast(castPoint.position, endPos, 1
<< LayerMask.NameToLayer("Player"));

            if (hit.collider != null)
            {
```

```
            if (hit.collider.gameObject.CompareTag("Player"))
            {
                val = true;
                ShotPlayer();
            }
            else
            {
                val = false;
            }
            Debug.DrawLine(castPoint.position, hit.point, Color.yellow);
        }
        else
        {
            Debug.DrawLine(castPoint.position, endPos, Color.blue);
        }
        return val;
    }
}
void ChasePlayer()
{
    if (transform.position.x < player.position.x)
    {
        rb2d.velocity = new Vector2(ms, 0);
        healthBarEnemy.localEulerAngles = Vector2.up * 180f;
        transform.localScale = new Vector2(0.6285622f, 0.6285622f);
        isFacingright = false;

    }
    else
    {
        rb2d.velocity = new Vector2(-ms, 0);
        healthBarEnemy.localEulerAngles = Vector3.zero;
        transform.localScale = new Vector2(-0.6285622f, 0.6285622f);
        isFacingright = true;
    }

}
public void Patrol()
{
    if (isFacingright == true)
    {
        rb2d.velocity = new Vector2(3.5f, 0);
        transform.localScale = new Vector2(0.6285622f, 0.6285622f);
    }


    if (isFacingright == false)
    {
```

```
            rb2d.velocity = new Vector2(-3.5f, 0);
            transform.localScale = new Vector2(-0.6285622f, 0.6285622f);
        }

        if (transform.position.x >= firstPos.x + 2)
        {
            isFacingright = false;
        }

        if (transform.position.x <= firstPos.x + (-2))
        {
            isFacingright = true;
        }
    }
    void ShotPlayer()
    {
        if (transform.position.x < player.position.x)
        {
            if (currentTimeSpawn > 0)
            {
                currentTimeSpawn -= Time.deltaTime;
            }
            else
            {

                Instantiate(bulletEnemy, shotPointEnemy.position,
transform.rotation);
                currentTimeSpawn = timeToSpawn;

            }

        }
        else
        {
            if (currentTimeSpawn > 0)
            {
                currentTimeSpawn -= Time.deltaTime;
            }
            else
            {
                Instantiate(bulletEnemy, shotPointEnemy.position,
transform.rotation);
                currentTimeSpawn = timeToSpawn;
            }

        }
    }
```

```
    void stopChasePlayer()
    {
        isAgro = false;
        isSearching = false;
        rb2d.velocity = new Vector2(0, 0);
    }
     }
```

2. Healt Enemy (Nyawa Musuh)

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

public class EnemyHealth : MonoBehaviour
{

    public float eHealt;
    public Image healthUI;
    public float maxHealth;

    public GameObject uiGems;

    public static EnemyHealth vc;

    private void Awake()
    {
        vc = this;
    }
    private void Start()
    {
        maxHealth = eHealt;
    }
    public void takeDemage(float demage)
    {
        eHealt -= demage;
        healthUI.fillAmount = eHealt / maxHealth;
        if (eHealt < 0)
        {
            Destroy(gameObject);
            enemyDead();
        }
    }
    void enemyDead()
    {
        uiGems.SetActive(true);
    }
```

```
    }
```

3. Bullet Enemy (Peluru Enemy)

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

public class BulletEnemy : MonoBehaviour
{
    // Start is called before the first frame update
    public float speed, demage, destroyTime;

    private void Awake()
    {
        Destroy(gameObject, destroyTime);
    }
    void Update()
    {
        shootBullet();
    }

    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.CompareTag("Player"))
        {
            HealtController.healController.Takedemage();

            Destroy(gameObject);

        }
        else if (collision.CompareTag("Envirotment"))
        {
            Destroy(gameObject);
        }
    }
    void shootBullet()
    {
        transform.Translate(Vector2.left * speed * Time.deltaTime);
    }
     }
```

4. Pergerakan Player

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

public class PlayerController : MonoBehaviour
```

```
{
    // Start is called before the first frame update
    public float ms, jumpForce, scaleX;
    public bool jalanKiri,jalanKanan, jump;

    public float radius;
    public Transform groundChecker;
    public LayerMask whatIsGround;

    Rigidbody2D rb;
    Animator anim;

    private void Awake()
    {
        scaleX = transform.position.x;
        rb = GetComponent<Rigidbody2D>();
        anim = GetComponent<Animator>();
    }
    void Update()
    {
        Jump();
    }
    private void FixedUpdate()
    {
        Move();
    }
    void Move()
    {
        if (Input.GetKey(KeyCode.A) || (jalanKiri==true))
        {
            transform.Translate(Vector2.left * -ms * Time.deltaTime, Space.Self);
            anim.SetBool("Run",true);
            transform.eulerAngles = Vector2.up * -180;
        }
        else if (Input.GetKey(KeyCode.D)||(jalanKanan==true))
        {
            transform.Translate(Vector2.right * ms * Time.deltaTime, Space.Self);
            anim.SetBool("Run", true);
            transform.eulerAngles = Vector2.zero;
        }
        else
        {
            anim.SetBool("Run", false);
        }
    }
    void Jump()
    {
        if (Input.GetKeyDown(KeyCode.Space) && IsGround() ||(jump==true))
```

```
    {
        rb.velocity = Vector2.up * jumpForce;
        anim.SetBool("Jump", true);
    }
    else
    {
        anim.SetBool("Jump", false);
    }
}

public void MainLagi()
{
    Time.timeScale = 1;
    SceneManager.LoadScene(1);
}
public void MainMenu()
{
    Time.timeScale = 1;
    SceneManager.LoadScene(0);

}
private void OnTriggerEnter2D(Collider2D collision)
{
    if (collision.CompareTag("deadArea"))
    {
        Destroy(gameObject);
        SceneManager.LoadScene(1);
    }
    else if (collision.CompareTag("Gems"))
    {
        GoalManager.singleton.collectedGems();
        Destroy(collision.gameObject);
    }
    if (collision.CompareTag("Duri"))
    {
        Destroy(gameObject);
        SceneManager.LoadScene(1);
    }
    else if (collision.CompareTag("Goal"))
    {
        if (GoalManager.singleton.canEnter)
        {
            print("silahkan masuk");
            SceneManager.LoadScene(2);
        }
    }
}
bool IsGround()
```

```
    {
        return Physics2D.OverlapCircle(groundChecker.position, radius,
whatIsGround);
    }
    private void OnDrawGizmos()
    {
        Gizmos.DrawWireSphere(groundChecker.position, radius);
    }
     }
```

5. Healt Controller player

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

public class HealtController : MonoBehaviour
{
    // Start is called before the first frame update
    public int healt;
    public GameObject[] healtUI;
    public GameObject screenUI;
    public static HealtController healController;

    private void Awake()
    {
        healController = this;
    }

    public void Takedemage()
    {
        healt--;
        if (healt <= 0)
        {
            healt = 0;
            PlayerLose();


        }
        healtUI[healt].SetActive(false);
    }
    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.CompareTag("Enemy"))
        {
            Takedemage();
        }
```

```
    }
    void PlayerLose()
    {
        screenUI.SetActive(true);
        Time.timeScale = 0;
    }
     }
```

6. Coin dan gems Controller Player

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

public class CoinController : MonoBehaviour
{
    // Start is called before the first frame update
    public Text scoreUI;
    public int score;

    public Text scoreUIGems;
    public int scoregems;



    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.CompareTag("Coin"))
        {
            Destroy(collision.gameObject);
            score++;
            scoreUI.text = score.ToString();
        }
        if (collision.CompareTag("Gems"))
        {
            Destroy(collision.gameObject);
            scoregems++;
            scoreUIGems.text = scoregems.ToString();

        }
    }
}
```

7. Player Shoot

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

```
public class PlayerShot : MonoBehaviour
{
    // Start is called before the first frame update
    public GameObject bullet;
    public Transform shotPoint;
    Animator anim;

    private void Start()
    {
        anim = GetComponent<Animator>();
    }

    void Update()
    {
        Shot();
    }

    void Shot()
    {
        if (Input.GetKeyDown(KeyCode.C))
        {
            anim.SetBool("Shoot", true);
            AudioManager.myAudio.PlaySound(0);
            Instantiate(bullet, shotPoint.position, transform.rotation);

        }
        else
        {
            anim.SetBool("Shoot", false);
        }
    }
     }
```

8. Bullet Player

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

public class bulletcontrol : MonoBehaviour
{
    // Start is called before the first frame update
    public float speed, demage, destroyTime;

    private void Awake()
    {
        Destroy(gameObject, destroyTime);
    }
    void Update()
```

```
    {
        transform.Translate(Vector2.right * speed * Time.deltaTime);
    }

    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.CompareTag("Enemy"))
        {

collision.transform.parent.GetComponent<EnemyHealth>().takeDemage(dem
age);
            Destroy(gameObject);
        }
        else if(collision.CompareTag("Envirotment"))
        {
            Destroy(gameObject);
        }
    }
}
```

9. Camera Follow

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

public class CameraFollow : MonoBehaviour
{

    public Transform target;
    public float smoothSpeed = 0.125f;


    public Vector3 offset;

    // Update is called once per frame
    void LateUpdate()
    {
        Vector3 desiredPosition = target.position + offset;
        Vector3 smoothedPosition = Vector3.Lerp(transform.position,
desiredPosition, smoothSpeed);
        transform.position = smoothedPosition;
    }
}
```

10. Rintangan Bounce Ball

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

```
public class DirtyBubblesControlls : MonoBehaviour
{
    Rigidbody2D rb;
    void Start()
    {
        rb = GetComponent<Rigidbody2D>();
    }

    // Update is called once per frame
    void Update()
    {
        transform.Translate(Vector2.up * Random.Range(8,12.8f) *
Time.deltaTime, Space.Self);
    }
}
```

11. Rintangan Tower Monster

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

public class TowerMonster : MonoBehaviour
{
    [SerializeField]
    Transform player;

    [SerializeField]
    Transform raycastPoint;

    [SerializeField]
    float agroRange;

    bool isAgro = false;
    bool isSearching = false;

    public GameObject bulletEnemy;
    public Transform shotPointEnemy;

    private float currentTimeSpawn;
    public float timeToSpawn;

    public Transform healthBarEnemy;

    public void Awake()
    {
```

```
    }

    // Update is called once per frame
    void Update()
    {
        if (CanSeePlayer(agroRange))
        {
            isAgro = true;
        }
        else
        {
            if (isAgro)
            {
                if (!isSearching)
                {
                    isSearching = true;
                    Invoke("stopChasePlayer", 2);
                }
            }

        }

        bool CanSeePlayer(float distance)
        {
            bool val = false;
            float castDist = distance;

            Vector2 endPos = raycastPoint.position + Vector3.left * castDist;

            RaycastHit2D hit = Physics2D.Linecast(raycastPoint.position, endPos,
1 << LayerMask.NameToLayer("Player"));

            if (hit.collider != null)
            {
                if (hit.collider.gameObject.CompareTag("Player"))
                {
                    ShotPlayer();
                    val = true;
                }
                else
                {
                    val = false;
                }
                Debug.DrawLine(raycastPoint.position, hit.point, Color.yellow);
            }
            else
            {
                Debug.DrawLine(raycastPoint.position, endPos, Color.blue);
```

```
            }
            return val;
        }
    }
    void ShotPlayer()
    {
        if (transform.position.x < player.position.x)
        {
            if (currentTimeSpawn > 0)
            {
                currentTimeSpawn -= Time.deltaTime;
            }
            else
            {
                Instantiate(bulletEnemy, shotPointEnemy.position,
transform.rotation);
                currentTimeSpawn = Random.RandomRange(1f,3f);
            }

        }
        else
        {
            if (currentTimeSpawn > 0)
            {
                currentTimeSpawn -= Time.deltaTime;
            }
            else
            {

                Instantiate(bulletEnemy, shotPointEnemy.position,
transform.rotation);
                currentTimeSpawn = Random.RandomRange(1f, 3f);
            }

        }
    }

    void stopChasePlayer()
    {
        isAgro = false;
    }
     }
```

12. Tower Bullet

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

```
public class towerBullet : MonoBehaviour
{
    // Start is called before the first frame update
    public float speedt, demaget, destroyTimet;

    private void Awake()
    {
        Destroy(gameObject, destroyTimet);
    }
    void Update()
    {
        transform.Translate(Vector2.right * speedt * Time.deltaTime);
    }

    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.CompareTag("Player"))
        {
            HealtController.healController.Takedemage();
            Destroy(gameObject);
            print("kena");
        }
        else if (collision.CompareTag("Envirotment"))
        {
            Destroy(gameObject);
        }
    }
     }
```

13. Balok Deteksi

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

public class kotakDetected : MonoBehaviour
{
    // Start is called before the first frame update
    Animator animm;

    private void Start()
    {
        animm = GetComponent<Animator>();
    }
    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.CompareTag("Detect"))
```

```
            {
                kotakpuzzle.iniKotakPuzzle.PuzzleFinished();
                Destroy(collision.gameObject);
            }else if (collision.CompareTag("lastDetect"))
            {
                if (kotakpuzzle.iniKotakPuzzle.canEnterGoa)
                {
                    print("Silahkan masuk");
                    BalokSlide.inibalokSlide.animasiBalok();
                }
            }
        }

    }
```

14. Balok Slide

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

public class BalokSlide : MonoBehaviour
{
    public static BalokSlide inibalokSlide;
    Animator anim;

    private void Awake()
    {
        inibalokSlide = this;
    }
    void Start()
    {
        anim = GetComponent<Animator>();
    }

    // Update is called once per frame
    void Update()
    {
    }
    public void animasiBalok()
    {
        anim.SetTrigger("balokSlide");

    }
}
```

15. Goal Controller

```
using System.Collections;
```

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

public class GoalManager : MonoBehaviour
{
    public static GoalManager singleton;

    public int gemsneeded;
    public int gemscollected;
    public bool canEnter;

    private void Awake()
    {
        singleton = this;
    }

    public void collectedGems()
    {
        gemscollected++;
        if(gemscollected >= gemsneeded)
        {
            canEnter = true;
        }
    }
}
```

16. Main Menu Controller

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

public class MainMenuController : MonoBehaviour
{
    /
    public void StartScene()
    {
        SceneManager.LoadScene(1);
    }
    public void StartAbout()
    {
        SceneManager.LoadScene(3);
    }
    public void AboutBack()
    {
        SceneManager.LoadScene(0);
    }
```

```
}
```

17. Floating Button

```
void PlayerLose()
{
  screenUI.SetActive(true);
  Time.timeScale = 0;
}
```

18. Link

Penjelasan Bagaimana cara rancang bangun dan vidio *gameplay game Ninja Adventure.*

Link : https://youtu.be/XkTuSbmtiqo

19. Backsound dalam game

Backsound dalam game dapat didownload secara free pada web Chosic. Judul backsound yang digunakan adalah Hostiles Inbound

Dapat diakses : hostiles Music Free Download MP3 - Chosic