1. Сценарий управления игроком

// PlayerMovement.cs

using UnityEngine;

public class PlayerMovement : MonoBehaviour

{

public float moveSpeed = 5f;

public float rotationSpeed = 720f;

private void Update()

{

// Input

float horizontal = Input.GetAxis("Horizontal");

float vertical = Input.GetAxis("Vertical");

// Movement

Vector3 direction = new Vector3(horizontal, 0, vertical).normalized;

transform.Translate(direction \* moveSpeed \* Time.deltaTime, Space.World);

// Rotation

if (direction.magnitude >= 0.1f)

{

float targetAngle = Mathf.Atan2(direction.x, direction.z) \* Mathf.Rad2Deg;

float angle = Mathf.SmoothDampAngle(transform.eulerAngles.y, targetAngle, ref rotationSpeed, 0.1f);

transform.rotation = Quaternion.Euler(0, angle, 0);

}

}

}

2. Сценарий управления бонусами

// BonusController.cs

using UnityEngine;

public class BonusController : MonoBehaviour

{

private Animator animator;

private void Start()

{

animator = GetComponent<Animator>();

}

private void OnTriggerEnter(Collider other)

{

if (other.CompareTag("Player"))

{

animator.SetTrigger("Enter");

}

}

private void OnTriggerExit(Collider other)

{

if (other.CompareTag("Player"))

{

animator.SetTrigger("Exit");

}

}

}

3. Сценарий для монеток и кошелька

// CoinController.cs

using UnityEngine;

public class CoinController : MonoBehaviour

{

public int coinValue = 1;

private void OnTriggerEnter(Collider other)

{

if (other.CompareTag("Player"))

{

Wallet.Instance.AddCoins(coinValue);

Destroy(gameObject);

}

}

}

// Wallet.cs

using UnityEngine;

public class Wallet : MonoBehaviour

{

public static Wallet Instance { get; private set; }

private int totalCoins = 0;

private void Awake()

{

if (Instance == null)

{

Instance = this;

DontDestroyOnLoad(gameObject);

}

else

{

Destroy(gameObject);

}

}

public void AddCoins(int amount)

{

totalCoins += amount;

Debug.Log("Total Coins: " + totalCoins);

}

}

4. Сценарий для взрывчатых бочек

// ExplosiveBarrel.cs

using UnityEngine;

public class ExplosiveBarrel : MonoBehaviour

{

public float explosionForceThreshold = 5f;

public GameObject explosionEffect; // эффект взрыва

private void OnCollisionEnter(Collision collision)

{

if (collision.gameObject.CompareTag("Player"))

{

if (collision.relativeVelocity.magnitude > explosionForceThreshold)

{

Instantiate(explosionEffect, transform.position, Quaternion.identity);

Destroy(collision.gameObject);

Destroy(gameObject);

}

}

}

}

5. Сценарий для подвижных препятствий

// MovingObstacle.cs

using UnityEngine;

public class MovingObstacle : MonoBehaviour

{

public float damageAmount = 10f;

public Vector3 moveDirection = Vector3.right;

public float moveSpeed = 2f;

public float moveDistance = 3f;

private Vector3 startPosition;

private float moveTimer;

private void Start()

{

startPosition = transform.position;

}

private void Update()

{

moveTimer += Time.deltaTime;

transform.position = startPosition + moveDirection \* Mathf.Sin(moveTimer \* moveSpeed) \* moveDistance;

}

private void OnTriggerEnter(Collider other)

{

if (other.CompareTag("Player"))

{

PlayerHealth playerHealth = other.GetComponent<PlayerHealth>();

if (playerHealth != null)

{

playerHealth.TakeDamage(damageAmount);

}

}

}

}

6. Сценарий для системы здоровья игрока

// PlayerHealth.cs

using UnityEngine;

public class PlayerHealth : MonoBehaviour

{

public float maxHealth = 100f;

private float currentHealth;

private void Start()

{

currentHealth = maxHealth;

}

public void TakeDamage(float amount)

{

currentHealth -= amount;

Debug.Log("Current Health: " + currentHealth);

if (currentHealth <= 0)

{

Die();

}

}

private void Die()

{

Debug.Log("Player Died!");

Destroy(gameObject);

}

}

7. Система частиц для игрока

// PlayerParticleController.cs

using UnityEngine;

public class PlayerParticleController : MonoBehaviour

{

private ParticleSystem particleSystem;

private void Start()

{

particleSystem = GetComponent<ParticleSystem>();

}

private void Update()

{

if (Input.GetAxis("Vertical") != 0 || Input.GetAxis("Horizontal") != 0)

{

if (!particleSystem.isPlaying)

{

particleSystem.Play();

}

}

else

{

if (particleSystem.isPlaying)

{

particleSystem.Stop();

}

}

}

}