Load next scene

using UnityEngine.SceneManagement;

public class SceneLoader : MonoBehaviour {

public void LoadNextScene()

{

int currentSceneIndex = SceneManager.GetActiveScene().buildIndex;

Debug.Log("Currscene = " + currentSceneIndex);

SceneManager.LoadScene(currentSceneIndex + 1);

}

public void LoadStartScene()

{

SceneManager.LoadScene(0);

}

}

Guess number

using UnityEngine.UI;

public class NumberWizard : MonoBehaviour {

[SerializeField] int max;

[SerializeField] int min;

int guess;

[SerializeField] Text text;

// Use this for initialization

void Start () {

StartGame();

}

void StartGame()

{

max = max + 1;

guess = (min + max) / 2;

text.text = guess.ToString();

}

public void nextGuess()

{

max = max + 1;

guess = (min + max) / 2;

text.text = guess.ToString();

}

public void onPressHigher()

{

min = guess;

nextGuess();

}

public void onPressLower()

{

max = guess;

nextGuess();

}

Padle following to mouse (horizontal only), range[minX, maxX]

using UnityEngine;

public class Paddle : MonoBehaviour {

[SerializeField] float screenWidthInUnits=16f;

[SerializeField] float minX = 1f;

[SerializeField] float maxX = 15f;

// Use this for initialization

void Start () {

}

// Update is called once per frame

void Update () {

float mousePosUnits = Input.mousePosition.x / Screen.width \* screenWidthInUnits;

Vector2 paddlePos = new Vector2(transform.position.x, transform.position.y);

paddlePos.x = Mathf.Clamp(mousePosUnits, minX, maxX);

transform.position = paddlePos;

}

}

Plane moves freedom(left, right, up, down)

public float planeSpeed;

Rigidbody2D myBody;

void Awake()

{

myBody = GetComponent<Rigidbody2D>();

}

void FixedUpdate()

{

planeMovement();

}

void planeMovement()

{

float xAxis = Input.GetAxisRaw("Horizontal") \* planeSpeed;

float yAxis = Input.GetAxisRaw("Vertical") \* planeSpeed;

myBody.velocity = new Vector2(xAxis, yAxis);

}

Thực hiện hành động sau một khoảng trễ thời gian

Sử dụng StartCoroutine(Ienumerator iEnumerator);

Vd: bắn ra 1 viên đạn từ vị trí hiện tại của máy bay:

[SerializeField]

private GameObject bullet;

private bool canShoot=true;

void Update () {

if (Input.GetKey(KeyCode.Space))

{

if (canShoot)

{

StartCoroutine(Shoot());

}

}

if (Input.GetMouseButton(0))

{

if(canShoot)

{

StartCoroutine(Shoot());

}

}

}

IEnumerator Shoot()

{

canShoot = false;

Vector3 tempt = transform.position;

//tempt.y += 0.6f;

//tạo clone của một đối tượng, bullet

Instantiate(bullet, tempt, Quaternion.identity);

yield return new WaitForSeconds(0.2f);

canShoot = true;

}

Rơi theo chiều dọc

public class YelloBullet : MonoBehaviour {

public float speed;

private Rigidbody2D myBody;

void Awake()

{

myBody = GetComponent<Rigidbody2D>();

}

void FixedUpdate()

{

myBody.velocity = new Vector2(0f, speed);

}

}

Scale backgroud equal to screen

public class BGScaleer : MonoBehaviour {

// Use this for initialization

void Start () {

SpriteRenderer sr = GetComponent<SpriteRenderer>();

Vector3 tempScale = transform.localScale;

float height = sr.bounds.size.y;

float width = sr.bounds.size.x;

float worldHeight = Camera.main.orthographicSize \* 2f;

float worldWidth = worldHeight \* Screen.width / Screen.height;

transform.localScale = new Vector3(worldWidth, worldHeight, 0);

tempScale.y = worldHeight / height;

tempScale.x = worldWidth / width;

transform.localScale = tempScale;

}

// Update is called once per frame

void Update () {

}

}

Cho pige chạy lui về sau

public class PigeHolder : MonoBehaviour

{

public float speed;

// Update is called once per frame

void Update()

{

\_PigeMovement();

}

void \_PigeMovement()

{

Vector3 temp = transform.position;

temp.x -= speed \* Time.deltaTime;

transform.position = temp;

}

}

Sinh pige sau một khoảng tg

public class SpawnerPige : MonoBehaviour {

[SerializeField]

private GameObject PigeHolder;

// Use this for initialization

void Start () {

StartCoroutine(Spawner());

}

IEnumerator Spawner()

{

yield return new WaitForSeconds(1.5f);

Vector3 temp = PigeHolder.transform.position;

temp.y = Random.Range(-1.8f, 2.2f);

Instantiate(PigeHolder, temp, Quaternion.identity);

StartCoroutine(Spawner());

}

}

BẮt va chạm có các hàm:

Void **OnCollisionEnter2D**(Collision2D target){

//được sử dụng khi cả 2 đối tượng đều không đc check isTrigger

}

Và

Void **OnTriggerEnter2D**(Collider target){

//được sử dụng khi một trong 2 đối tượng có check istrigger

if (target.tag == "Destroy")//gắn tag Destroy cho target

{

Destroy(gameObject);

}

}

Để 2 đối tượng có thể va chạm được thì :

1/ Cả 2 đều phải có Collider

2/ Một trong 2 phải có Regidbody

Chim nhảy khi click chuột

public float bounceForce;

private Rigidbody2D mybody;

void Awake()

{

mybody = GetComponent<Rigidbody2D>();

}

void FixedUpdate()

{

\_BirdMovement();

}

void \_BirdMovement()

{

If(Input.getMouseButton(0)

{

mybody.velocity = new Vector2(mybody.velocity.x, bounceForce);

} }