Shiela Marie Flojo

BSIT A703

**DO Jump Source Code**

**PlayerController.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

using Cinemachine;

[RequireComponent(typeof(Rigidbody2D))]

[RequireComponent(typeof(CapsuleCollider2D))]

public class PlayerController : MonoBehaviour

{

int gemCount=0;

int peopleCount=0;

// Move player in 2D space

public float maxSpeed = 3.4f;

public float jumpHeight = 6.5f;

public float gravityScale = 1.5f;

bool facingRight = true;

float moveDirection = 0;

bool isGrounded = true;

Vector3 cameraPos;

Rigidbody2D r2d;

CapsuleCollider2D mainCollider;

Transform t;

SpriteRenderer spriteRenderer;

public Sprite newSprite;

public AudioSource audioSourceDiamond;

public AudioSource audioSourcePeople;

public Text diamondUICount;

public Text peopleUICount;

public CinemachineVirtualCamera mainCamera;

public CinemachineVirtualCamera endCamera;

public Canvas endDialogueCard;

bool mobileDevice = true;

public GameObject canvas;

public Joystick joyStickPrefab;

void Awake(){

if (!Application.isMobilePlatform)

{

mobileDevice=false;

joyStickPrefab.gameObject.SetActive(false);

}

}

void Start()

{

t = transform;

r2d = GetComponent<Rigidbody2D>();

mainCollider = GetComponent<CapsuleCollider2D>();

r2d.freezeRotation = true;

r2d.collisionDetectionMode = CollisionDetectionMode2D.Continuous;

r2d.gravityScale = gravityScale;

facingRight = t.localScale.x > 0;

}

// Update is called once per frame

void Update()

{

// Movement controls on mobile device

if (mobileDevice){

if ((joyStickPrefab.Horizontal >= .2f) && (isGrounded || Mathf.Abs(r2d.velocity.x) > 0.01f)){

moveDirection=1;

}

else{

if((joyStickPrefab.Horizontal <= -.2f) && (isGrounded || Mathf.Abs(r2d.velocity.x) > 0.01f)){

moveDirection=-1;

}else{

moveDirection=0;

}

}

// Jumping

if ( (joyStickPrefab.Vertical>= .5f) && isGrounded)

{

r2d.velocity = new Vector2(r2d.velocity.x, jumpHeight);

}

}

// Movement controls on other devices

else{

if ((Input.GetKey(KeyCode.RightArrow) || Input.GetKey(KeyCode.LeftArrow)) && (isGrounded || Mathf.Abs(r2d.velocity.x) > 0.01f))

{

moveDirection = Input.GetKey(KeyCode.LeftArrow) ? -1 : 1;

}else{

if (isGrounded || r2d.velocity.magnitude < 0.01f)

{

moveDirection = 0;

}

}

// Jumping

if (Input.GetKeyDown(KeyCode.UpArrow) && isGrounded)

{

r2d.velocity = new Vector2(r2d.velocity.x, jumpHeight);

}

}

// Change facing direction

if (moveDirection != 0)

{

if (moveDirection > 0 && !facingRight)

{

facingRight = true;

t.localScale = new Vector3(Mathf.Abs(t.localScale.x), t.localScale.y, transform.localScale.z);

}

if (moveDirection < 0 && facingRight)

{

facingRight = false;

t.localScale = new Vector3(-Mathf.Abs(t.localScale.x), t.localScale.y, t.localScale.z);

}

}

}

void FixedUpdate()

{

Bounds colliderBounds = mainCollider.bounds;

float colliderRadius = mainCollider.size.x \* 0.4f \* Mathf.Abs(transform.localScale.x);

Vector3 groundCheckPos = colliderBounds.min + new Vector3(colliderBounds.size.x \* 0.5f, colliderRadius \* 0.9f, 0);

// Check if player is grounded

Collider2D[] colliders = Physics2D.OverlapCircleAll(groundCheckPos, colliderRadius);

//Check if any of the overlapping colliders are not player collider, if so, set isGrounded to true

isGrounded = false;

if (colliders.Length > 0)

{

for (int i = 0; i < colliders.Length; i++)

{

if (colliders[i] != mainCollider)

{

isGrounded = true;

break;

}

}

}

// Apply movement velocity

r2d.velocity = new Vector2((moveDirection) \* maxSpeed, r2d.velocity.y);

}

void OnTriggerEnter2D(Collider2D col) {

if (col.gameObject.tag == "diamond") {

gemCount+=1;

audioSourceDiamond.Play();

UpdateDiamondCountUI();

Destroy (col.gameObject);

}

if (col.gameObject.tag == "sad" && gemCount>0) {

spriteRenderer = col.gameObject.GetComponent<SpriteRenderer>();

spriteRenderer.sprite = newSprite;

audioSourcePeople.Play();

gemCount-=1;

peopleCount+=1;

UpdatePeopleCountUI();

UpdateDiamondCountUI();

col.gameObject.tag="happy";

}

if (col.gameObject.tag == "end") {

endDialogueCard.transform.GetChild(0).gameObject.SetActive(true);

endCamera.gameObject.SetActive(true);

mainCamera.gameObject.SetActive(false);

}

}

void UpdateDiamondCountUI(){

diamondUICount.text=gemCount.ToString();

}

void UpdatePeopleCountUI(){

peopleUICount.text=peopleCount.ToString();

}

}

**Dialogue.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

[System.Serializable]

public class Dialogue {

public string name;

[TextArea(3, 10)]

public string[] sentences;

}

**DialogueManager.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

using UnityEngine.SceneManagement;

using Cinemachine;

public class DialogueManager : MonoBehaviour {

public Text nameText;

public Text dialogueText;

public Dialogue dialogue;

public string sceneName;

//public Animator animator;

private Queue<string> sentences;

int dialogueCount=-1;

public CinemachineVirtualCamera[] cameras;

public CinemachineVirtualCamera homeCamera;

public Canvas dialogueCard;

// Use this for initialization

void Start () {

sentences = new Queue<string>();

StartDialogue(dialogue);

//animator.SetBool("IsOpen", false);

}

public void StartDialogue (Dialogue dialogue)

{

// animator.SetBool("IsOpen", true);

nameText.text = dialogue.name;

sentences.Clear();

foreach (string sentence in dialogue.sentences)

{

sentences.Enqueue(sentence);

}

DisplayNextSentence();

}

public void DisplayNextSentence ()

{

if (sentences.Count == 0)

{

EndDialogue();

return;

}

print(cameras[dialogueCount+1]);

changeCamera();

string sentence = sentences.Dequeue();

StopAllCoroutines();

StartCoroutine(TypeSentence(sentence));

}

IEnumerator TypeSentence (string sentence)

{

dialogueText.text = "";

foreach (char letter in sentence.ToCharArray())

{

dialogueText.text += letter;

yield return null;

}

}

void EndDialogue()

{

if (dialogueCard !=null){

dialogueCard.transform.GetChild(0).gameObject.SetActive(false);

//.GetComponent<SpriteRenderer>().enabled = false;

//dialogueCard.enabled=false;

homeCamera.gameObject.SetActive(true);

cameras[cameras.Length - 1].gameObject.SetActive(false);

}

}

void changeCamera(){

if (cameras!=null || homeCamera!=null){

if (dialogueCount>=0){

cameras[dialogueCount+1].gameObject.SetActive(true);

cameras[dialogueCount].gameObject.SetActive(false);

}

dialogueCount+=1;

}

}

}

**IntroManager.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

using UnityEngine.SceneManagement;

using Cinemachine;

public class IntroManager : MonoBehaviour {

public Text nameText;

public Text dialogueText;

public Dialogue dialogue;

public string sceneName;

// sentences for dialogue

private Queue<string> sentences;

// Use this for initialization

void Start () {

sentences = new Queue<string>();

StartDialogue(dialogue);

}

public void StartDialogue (Dialogue dialogue)

{

nameText.text = "Screen Smiles: The Power of Online Positivity!";

sentences.Clear();

foreach (string sentence in dialogue.sentences)

{

sentences.Enqueue(sentence);

}

DisplayNextSentence();

}

public void DisplayNextSentence ()

{

if (sentences.Count == 0)

{

EndDialogue();

return;

}

string sentence = sentences.Dequeue();

StopAllCoroutines();

StartCoroutine(TypeSentence(sentence));

}

IEnumerator TypeSentence (string sentence)

{

dialogueText.text = "";

foreach (char letter in sentence.ToCharArray())

{

dialogueText.text += letter;

yield return null;

}

}

void EndDialogue()

{

SceneManager.LoadScene(sceneName);

}

}

**LoadScene.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

public class LoadScene : MonoBehaviour

{

public string SceneName;

public void loadScene(string SceneName)

{

SceneManager.LoadScene(SceneName);

}

}