Unity code from group 406

Indhold

PostWwiseEvent.cs

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

public class PostWwiseEvent : MonoBehaviour
{
    public AK.Wwise.Event myEvent;
    // Start is called before the first frame update
    public void PlayFootStep()
    {
        myEvent.Post(gameObject);
    }

    // Update is called once per frame
    void Update()
    {

    }
}
```

movementScript.cs

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


public class movementScript : MonoBehaviour
{

    public float movementSpeed;
    public Vector3 movement;
    Rigidbody rb;
    public float distanceCounter;
    public float timeCounter;
    public bool ended;
    public string username;
    private SimpleCharacterControlFree controller;
    public GameObject manager;



    // Start is called before the first frame update
    void Start()
    {
        rb = this.gameObject.GetComponent<Rigidbody>();
        username =
GameObject.Find("userNameHolder").GetComponent<generateUserName>().username;


        controller = gameObject.GetComponent<SimpleCharacterControlFree>();
    }

    private void Update()
    {
        //movement = new Vector3(Input.GetAxis("Horizontal"), 0, Input.GetAxis("Vertical"));
```

```
        if (distanceCounter > 0 && ended != true) timeCounter += Time.deltaTime;

        if (controller.m_currentV > .01 || controller.m_currentH > 0.01 ||
controller.m_currentV < -.01 || controller.m_currentH <- 0.01)
        {
            //AkSoundEngine.PostEvent("FootStep", gameObject);
        }

    }

    void FixedUpdate()
    {
        //    moveCharacter(movement);

        //}

        //void moveCharacter(Vector3 direction)
        //{
        //    rb.MovePosition((Vector3)transform.position + (direction * movementSpeed *
Time.deltaTime));

        //if (controller.velocity != Vector3.zero)
        //{
            distanceCounter += Mathf.Abs(controller.m_currentH +
controller.m_currentV)*Time.deltaTime;

        //}

    }

    void OnTriggerEnter(Collider other)
    {

        if (other.tag == "End")
        {
            ended = true;
            manager.GetComponent<Help>().End();
        }
    }
}
```

Logic.cs

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

public class Logic : MonoBehaviour
{
    public GameObject boxCollider;
    public float offset = 1.0f;
    public Vector3 start = Vector3.zero;
    public Vector3 trans = Vector3.zero;
    public List<GameObject> cells;
    public List<int> path_1_list;
    [TextArea(15, 20)]
    public string path_1_ids;
    public int lastID;
    public AK.Wwise.Event goodSound,badSound;
```

```
    private bool goodPlayed, badPlayed = true;



    public GameObject player;
    public Vector3 lastCorrect = Vector3.zero;

    void Start()
    {
        start = boxCollider.transform.position;
        trans = start;

        for (int i = 0; i < 40; i++)
        {
            Instantiate(boxCollider, new Vector3(trans.x, trans.y, trans.z + i * offset),
Quaternion.identity);
            for (int j = 0; j < 40; j++)
            {
                GameObject newCell = Instantiate(boxCollider, new Vector3(trans.x + j *
offset, trans.y, trans.z + i * offset), Quaternion.identity);
                cells.Add(newCell);
                newCell.GetComponent<Cell>().id = i + j * 40;
            }
        }
        path_1_list.AddRange(path_1_ids.Split(',').Select(i => int.Parse(i)));
    }

    public void PathCheck(int cellID, GameObject cellObject)
    {
        var index = path_1_list.IndexOf(lastID);

        if (path_1_list.Contains(cellID) && path_1_list.Contains(lastID) && cellID != 19 &&
cellID != 0)
        {
            if (path_1_list[index + 1] == cellID)
            {

                if (!goodPlayed)
                {
                    PlayGood();
                }
                goodPlayed = true;
                badPlayed = false;
            }

            else
            {

                if (!badPlayed)
                {
                    PlayBad();
                }
                goodPlayed = false;
                badPlayed = true;
            }
        }

        else if (!path_1_list.Contains(cellID) || !path_1_list.Contains(lastID))
        {
            if (!badPlayed)
```

```
        {
            PlayBad();
        }
        goodPlayed = false;
        badPlayed = true;

        CheckMinorChoices(cellID);
    }
}

private void CheckMinorChoices(int cell) {


    if (cell == 48 || cell == 89)//Wrong cells
    {
        if (lastID == 49)
        {
            PlayBad();
        }
    }
    else if(cell == 50 && lastID == 49)//Correct cells
    {
        PlayGood();
    }
    ////////////////////////////////////////////
    if ((cell == 1094 || cell == 1135) && lastID == 1095)//Wrong cells
    {
            PlayBad();
    }
    else if (cell == 1055 && lastID == 1095)//Correct cells
    {
        PlayGood();
    }
    ////////////////////////////////////////////
    if ((cell == 1094 || cell == 1135) && lastID == 1095)//Wrong cells
    {
        PlayBad();
    }
    else if (cell == 1055 && lastID == 1095)//Correct cells
    {
        PlayGood();
    }
    ////////////////////////////////////////////
    if (cell == 1539 && lastID == 1579)//Wrong cells
    {
        PlayBad();
    }
    else if ((cell == 1578 || cell == 1580) && lastID == 1579)//Correct cells
    {
        PlayGood();
    }
    ////////////////////////////////////////////
    if (cell == 1309 && lastID == 1269)//Wrong cells
    {
        PlayBad();
    }
    else if ((cell == 1268 || cell == 1229) && lastID == 1269)//Correct cells
    {
        PlayGood();
    }
```

```
//////////////////////////////////////////////
if ((cell == 1038 || cell == 1079) && lastID == 1039)//Wrong cells
{
    PlayBad();
}
else if (cell == 999 && lastID == 1039)//Correct cells
{
    PlayGood();
}
//////////////////////////////////////////////
if ((cell == 956 || cell == 954) && lastID == 955)//Wrong cells
{
    PlayBad();
}
else if (cell == 915 && lastID == 955)//Correct cells
{
    PlayGood();
}
//////////////////////////////////////////////
if ((cell == 279 || cell == 359) && lastID == 319)//Wrong cells
{
    PlayBad();
}
else if (cell == 318 && lastID == 319)//Correct cells
{
    PlayGood();
}
//else if (cell == 1095)
//{

//    localBadPlayed = false;

//    if (lastID == 1094 || lastID == 1135)
//    {
//        PlayGood();
//    }
//}

//else if (cell == 1579 && (lastID == 1539))
//{
//        PlayGood();
//    localBadPlayed = false;
//}

//else if (cell == 1269)
//{

//    localBadPlayed = false;

//    if (lastID == 1309)
//    {
//        PlayGood();
//    }
//}

//else if (cell == 1039)
//{
//    localBadPlayed = false;
//    if (lastID == 1079 || lastID == 1038)
//    {
```

```
        //        PlayGood();
        //    }
        //}

        //else if (cell == 955)
        //{
        //    localBadPlayed = false;
        //    if (lastID == 954 || lastID == 956)
        //    {
        //        PlayGood();
        //    }
        //}

        //else if (cell == 319)
        //{
        //    localBadPlayed = false;
        //    if (lastID == 279 || lastID == 359)
        //    {
        //        PlayGood();
        //    }
        //}

        //else
        //{
        //    if ((!localBadPlayed))
        //    {
        //        PlayBad();
        //        localBadPlayed = true;
        //    }
        //}
    }




    //else if (path_1_list.Contains(lastID) && !path_1_list.Contains(cellID) && cellID !=
19)
    //{
    //    print("Incorrect");
    //    badSound.Post(gameObject);
    //}

    //else
    //{
    //    if (path_1_list[index] != cellID)
    //    {
    //        badSound.Post(gameObject);
    //    }
    //}

    //if (path_1_list.Contains(cellID))
    //{
    //    if (path_1_list[index] == cellID)
    //    {
    //        //print("correct");
    //        //goodSound.volume = 1.0f;
    //        //badSound.volume = 0.0f;
    //        goodSound.Post(gameObject);
    //        lastCorrect = cellObject.transform.position;
```

```
    //    }
    //    else
    //    {
    //        //print("incorrect");
    //        offset = Vector3.Distance(player.transform.position, lastCorrect);
    //        badSound.Post(gameObject);
    //        //goodSound.volume = 1 - (offset/10);
    //        //badSound.volume = offset / 10;
    //    }
    //}
    //else
    //{
    //    //print("incorrect");
    //    badSound.Post(gameObject);
    //    offset = (lastCorrect - player.transform.position).magnitude;
    //    //goodSound.volume = 1 - (offset / 10);
    //    //badSound.volume = offset / 10;
    //}



    private void PlayGood()
    {
        goodSound.Post(gameObject);
    }

    private void PlayBad()
    {
        badSound.Post(gameObject);
    }
}
```

Instructions.cs

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

public class Instructions : MonoBehaviour
{
    public Canvas canvas;
    // Phrygian scale = ominous
    void Start()
    {
        Invoke("ShowInstructions", 2);

    }

    private void ShowInstructions()
    {
        canvas.enabled = true;
        print("called");
    }

    public void CloseInstructions()
    {
        canvas.enabled = false;
    }
```

```
}
```

Help.cs

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

public class Help : MonoBehaviour
{

    public GameObject tutorial;
    public GameObject player;
    public GameObject EndScreen;
    public GameObject exitHelpButtons;
    public GameObject rageEndScreen;
    public movementScript playerData;
    public databaseSendData database;
    public Text[] UICode;
    int usernameInt;
    public Logic audioCues;
    public AkEvent orb;
    public string condition;
    private bool sentData;
    public GameObject codePanel;
    public Text codePanelText;
    void Start()
    {
        string username =
GameObject.Find("userNameHolder").GetComponent<generateUserName>().username;
        for (int i = 0; i < UICode.Length; i++)
        {
            UICode[i].text = "Code: " + username.ToString();
        }

        ShowInstructions();

        usernameInt = int.Parse(username);

        if (usernameInt % 2 == 1)
        {
            audioCues.enabled = false;
            orb.enabled = false;
            condition = "b";
        }

        else
        {
            condition = "a";
        }


    }

    public void ShowInstructions()
    {
```

```
    tutorial.SetActive(true);
    exitHelpButtons.SetActive(false);
    FreezePlayer();
}

public void CloseInstructions()
{
    tutorial.SetActive(false);
    exitHelpButtons.SetActive(true);
    UnfreezePlayer();
}

public void End()
{
    exitHelpButtons.SetActive(false);
    EndScreen.SetActive(true);
    FreezePlayer();
}

private void FreezePlayer()
{
    player.GetComponent<SimpleCharacterControlFree>().enabled = false;
    player.GetComponent<Animator>().enabled = false;
    player.GetComponent<Rigidbody>().isKinematic = true;
}

private void UnfreezePlayer()
{
    player.GetComponent<SimpleCharacterControlFree>().enabled = true;
    player.GetComponent<Animator>().enabled = true;
    player.GetComponent<Rigidbody>().isKinematic = false;
}

public void EndGame()
{
    print(usernameInt);
    if (usernameInt % 2 == 0) // A Test (Audio)
    {
        Application.OpenURL("https://forms.gle/YKvXj2NwqAsw24xT7");
        print("Option 1");
    }

    else // B Test (No Audio)
    {
        Application.OpenURL("https://forms.gle/XiNXVMwZpbaivMndA");
        print("Option 2");
    }

    print("quit");

    Screen.SetResolution(1280, 720, false);
    codePanelText.text = playerData.username;
    rageEndScreen.SetActive(false);
    codePanel.SetActive(true);

}

public void RageEndGame()
{
    playerData.ended = true;
```

```
        if (sentData == false)
        {
            StartCoroutine(database.Upload(playerData.username.ToString() + condition,
                                playerData.timeCounter.ToString(),
                                playerData.distanceCounter.ToString(),
                                playerData.ended.ToString()));
            sentData = true;
        }


        EndGame();
    }

    public void rageQuit()
    {
        FreezePlayer();
        rageEndScreen.SetActive(true);
        exitHelpButtons.SetActive(false);
    }

    public void CancelRageQuit()
    {
        playerData.ended = false;
        UnfreezePlayer();
        rageEndScreen.SetActive(false);
        exitHelpButtons.SetActive(true);
    }

}
```

GenerateUserName.cs

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


public class generateUserName : MonoBehaviour
{

    public string username;
    public int startvalue,endValue;
    public string[] usernamesFromDataBase;
    public GameObject usernameHolder;

    // Start is called before the first frame update

    private void Awake()
    {
        StartCoroutine(getText());
        DontDestroyOnLoad(this.gameObject);


    }


    private void Start()
```

```
    {
        generateANumber(startvalue,endValue);


    }
    IEnumerator getText()
    {
        UnityWebRequest www = UnityWebRequest.Get("http://switty.dk/nameCheck.php");
        yield return www.SendWebRequest();
        if (www.isNetworkError || www.isHttpError)
        {
            Debug.Log(www.error);
        }
        else
        {
            // Show results as text
            Debug.Log(www.downloadHandler.text);
            usernamesFromDataBase = www.downloadHandler.text.Split(':');

            // Or retrieve results as binary data
            byte[] results = www.downloadHandler.data;
        }
    }

    public void generateANumber(int minrange,int maxrange)
    {
        string newUsername = Random.Range(minrange,maxrange).ToString();

        for (int i = 0; i < usernamesFromDataBase.Length; i++)
        {
            if (newUsername == usernamesFromDataBase[i])
            {
                Debug.Log("error");
                generateANumber(minrange,maxrange);
            }

        }
        Debug.Log("new username has be generated");
        username = newUsername;
        usernameHolder.transform.GetChild(1).GetComponent<Text>().text = "Code: " +
username;

    }


}
```

followPlayer.cs

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

public class followPlayer : MonoBehaviour
{
    public GameObject player;       //Public variable to store a reference to the player
game object
    //private string playerTag = "Player";
```

```
    Vector3 offset;         //Private variable to store the offset distance between the
player and camera
    void Start()
    {
        //player = GameObject.FindGameObjectWithTag(playerTag);
        //Calculate and store the offset value by getting the distance between the player's
position and camera's position.
        offset = transform.position - player.transform.position;
    }

    // LateUpdate is called after Update each frame
    void Update()
    {
        // Set the position of the camera's transform to be the same as the player's, but
offset by the calculated offset distance.
        transform.position = player.transform.position + offset;
    }
}
```

databaseSendData.cs

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

public class databaseSendData : MonoBehaviour
{
    // Start is called before the first frame update
    string timeSpent;
    string distanceTraveled;
    string username;
    string rageQuit;
    bool sendData = false;
    public Text usernameHolder;
    public Help help;


    private void OnTriggerEnter(Collider other)
    {
        if (other.tag == "Player" && sendData == false)
        {
            username =
GameObject.Find("userNameHolder").GetComponent<generateUserName>().username;
            usernameHolder.text ="Code: "+ username;
            timeSpent =  other.GetComponent<movementScript>().timeCounter.ToString();
            distanceTraveled =
other.GetComponent<movementScript>().distanceCounter.ToString();
            rageQuit = other.GetComponent<movementScript>().ended.ToString();
            StartCoroutine(Upload(username + help.condition, timeSpent,
distanceTraveled,rageQuit));
            Debug.Log("timer :" + timeSpent);
            Debug.Log("distance :" + distanceTraveled);
            Debug.Log("username : "+username);
            sendData = true;
        }
    }
```

```
    public IEnumerator Upload(string playerName, string timer,string distanceCounter,string
rageQuit)
    {
        WWWForm form = new WWWForm();
        form.AddField("playerName", playerName);
        form.AddField("timer", timer);
        form.AddField("distance", distanceCounter);
        form.AddField("rageQuit", rageQuit);


        UnityWebRequest www = UnityWebRequest.Post("http://switty.dk/SendData.php", form);
        yield return www.SendWebRequest();

        if (www.isNetworkError || www.isHttpError)
        {
            Debug.Log(www.error);
        }
        else
        {
            Debug.Log("Form upload complete!");
        }
    }

}
```

Correct.cs

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

public class Correct : MonoBehaviour
{
    public AudioSource audioSource;
    public float rate;
    public bool entered;
    public int counter;
    public int mod;
    private collisionChecker leftCol, rightCol, backCol, forwardCol;


    private void Start()
    {
        counter = 0;
        leftCol = transform.Find("Left").GetComponent<collisionChecker>();
        rightCol = transform.Find("Right").GetComponent<collisionChecker>();
        forwardCol = transform.Find("Forward").GetComponent<collisionChecker>();
        backCol = transform.Find("Back").GetComponent<collisionChecker>();


    }

    public enum Karma
    { Correct, Incorrect }

    public enum Direction
    { Left, Right, Forward, Back }
```

```
    public Karma karma;
    public Direction direction;

    public void OnTriggerEnter(Collider collision)
    {
        counter +=  1;
        mod = counter % 2;
        entered = true;

        //if (karma == Karma.Incorrect && mod == 1)
        //{
        //    audioSource.volume += 0.1f;

        //}

        //if (karma == Karma.Incorrect && mod == 0)
        //{
        //    audioSource.volume -= 0.1f;
        //}

        //entered = true;

        //if (karma == Karma.Correct && mod == 1)
        //{
        //    audioSource.volume -= 0.1f;
        //}

        //if (karma == Karma.Correct && mod == 0)
        //{
        //    audioSource.volume += 0.1f;
        //}
    }


    public void Update()
    {
        if (entered == true)
        {
            if ((leftCol.hit && direction == Direction.Left)|| (rightCol.hit && direction ==
Direction.Right) || (forwardCol.hit && direction == Direction.Forward) || (backCol.hit &&
direction == Direction.Back))
            {
                audioSource.pitch = Mathf.Lerp(audioSource.pitch, 1.0f, rate);
                audioSource.volume = Mathf.Lerp(audioSource.volume, 0.5f, rate);
                //audioSource.volume = 0.0f;
            }

            if ((leftCol.hit && direction != Direction.Left) || (rightCol.hit && direction
!= Direction.Right) || (forwardCol.hit && direction != Direction.Forward) || (backCol.hit &&
direction != Direction.Back))
            {
                audioSource.pitch = Mathf.Lerp(audioSource.pitch, 1.45f, rate);
                audioSource.volume = Mathf.Lerp(audioSource.volume, 1.0f, rate);
            }



        }
```

```
    }
    //public void Update()
    //{
    //    if (entered == true)
    //    {
    //        if (karma == Karma.Correct && mod == 0)
    //        {
    //            audioSource.pitch = Mathf.Lerp(audioSource.pitch, 2.0f, rate);

    //        }

    //        if (karma == Karma.Correct && mod == 1)
    //        {
    //            audioSource.pitch = Mathf.Lerp(audioSource.pitch, 1.5f, rate);

    //        }
    //        if (karma == Karma.Incorrect && mod == 0)
    //        {
    //            audioSource.pitch = Mathf.Lerp(audioSource.pitch, 1.5f, rate);
    //        }

    //        if (karma == Karma.Incorrect && mod == 1)
    //        {
    //            audioSource.pitch = Mathf.Lerp(audioSource.pitch,   0.0f, rate);
    //        }

    //    }


    //}


}
```

collisionChecker.cs

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

public class collisionChecker : MonoBehaviour
{
    public bool hit;

    public void OnTriggerEnter(Collider collision)
    {
        if (collision.gameObject.CompareTag("Player"))
        {
            hit = true;
        }
    }

    public void OnTriggerExit(Collider collision)
    {
        if (collision.gameObject.CompareTag("Player"))
        {
            hit = false;
        }
    }
```

```
}
```

Cell.cs

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

public class Cell : MonoBehaviour
{

    public int id;
    public Logic logic;


    private void OnTriggerEnter(Collider other)
    {

        logic.PathCheck(id, this.gameObject);

    }

    private void OnTriggerExit(Collider other)
    {
        logic.lastID = id;
    }
}
```

buttonController.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;
using UnityEngine.UI;
public class buttonController : MonoBehaviour
{

    public GameObject[] buttonPanels;
    public Slider slider;
    private float sliderValue;


    private void Awake()
    {

        Screen.SetResolution(1920, 1080, true);
    }
    public void Update()
    {
        sliderValue = slider.value;
        AkSoundEngine.SetRTPCValue("PanSilder",sliderValue);



    }
```

```
    public void StartGame(int level)
    {
        AkSoundEngine.StopAll();
        Debug.Log("i was pressed");
        SceneManager.LoadScene(level);

    }

    public void Confirm()
    {
        slider.value = 0;
        buttonPanels[0].SetActive(false);
        buttonPanels[1].SetActive(true);

    }

    public void ExitGame()
    {
        Application.Quit();
    }

   // public void Help()
    //{
      //  ShowInstructions();
   // }


}
```

Cell.cs

arduinoController.cs

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

public class arduinoController : MonoBehaviour
{

    public float movementSpeed;
    public Vector3 movement;
    Rigidbody rb;
    public char direction;
    SerialPort sp = new SerialPort("COM3",9600);

    // Start is called before the first frame update
    void Start()
    {
        rb = this.gameObject.GetComponent<Rigidbody>();
        sp.Open();
        sp.ReadTimeout = 1;
    }

    private void Update()
    {

        if (sp.IsOpen)
        {
            try
            {
                direction = System.Convert.ToChar(sp.ReadByte());
                Debug.Log(direction);
            }
            catch (System.Exception)
            {
            }
        }
        if (direction =='W')
        {
            movement = new Vector3(0, 0, 1);
        }
        if (direction == 'S')
        {
            movement = new Vector3(0, 0, -1);
        }
        if (direction == 'A')
        {
            movement = new Vector3(-1, 0, 0);
        }
        if (direction == 'D')
        {
            movement = new Vector3(1, 0, 0);
        }

    }

    void FixedUpdate()
    {
        moveCharacter(movement);
```

```
    }

    void moveCharacter(Vector3 direction)
    {
        rb.MovePosition((Vector3)transform.position + (direction * movementSpeed *
Time.deltaTime));
    }
}
```