UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola d'Enginyeria de Barcelona Est

DEGREE DISERTATION

Industrial electronics and automation engineering degree

Electrical engineering degree

DESIGN OF VR APP APPLIED TO COGNITIVE TRAINING

Volume II

Authors: Isarn Ardanuy Cabezas
Xavier Riera Montell
Director: Jordi Torner Ribé
Co-Director: Francesc Alpiste Penalba

Date: Juny 2018

Annex A. Pilot Study Script

Hypothesis	The VR (Virtual reality) program was expected to achieve better encoding of information by the multimedia nature of the training, better recall of newly learned information by spaced retrieval, and motivation for persistent training through the subjective "presence" or the subjective feeling of being present in a simulated environment The two study objectives are (1) to develop and implement a non-immersive VR-based memory training program for older adults with questionable dementia, and (2) to examine the efficacy of a VR memory training program on episodic memory and self-appraisal of cognitive functions as compared with a conventional therapist-led memory training program. The study presents two hypotheses: (1) Positive changes would be exhibited in the VR-based and therapist-led memory training. (2) The VR-based memory group show better memory functions and greater functional independence as compared with the therapist-led group.		Aguinis et al., 2001 Man et al., 2012
Participants		40	
Control Group	Non-VR based memory programme	10	R. Sánchez proposed a
Experimental	An immersive VR- based memory training	10	

Group	programme		crossover study, i.e.: "relating to or denoting trials of medical treatment in which experimental subjects and control groups are exchanged after a set period."
Training content	An immersive VR- based memory training programme and a non-VR programme are developed using the same scenarios. Each scenario was designed according to a gradation structure: tasks ranged from simple to more complex in terms of the number and similarity of objects to be remembered as well as the duration of distraction. Each scenario was designed according to a gradation structure: tasks ranged from simple to more complex in terms of the number and similarity of objects to be remembered as well as the duration of distraction. The training was also upgraded systematically according to		Thivierge et al., 2008

	(a) duration of recall according to the spaced retrieval technique (Thivierge et al., 2008); (b) types of memory stimuli delivered: from visual-auditory and visual only to auditory only; and (c) migration from semantic to episodic memory, by requiring the participants to gradually remember what to do and the sequence of actions.		
Sessions number	20 patients (10 Control, 10 Experimental) Crossover. That means, finally all patients will use VR configuration. 24 individual sessions, 30 minutes/session 2 sessions/week 12 weeks/patient 12h/patient Technical and therapeutic support will be needed for the experiment in: "Centre de dia c/Reina Amalia, 37, al lado de la parada de metro Paralelo." 12h * 20 = 240h (2 mornings a week). Probable experiment duration: if 10h/week then 24 week (6 months)	24	
Methodology	Each session of the experimental and control training lasted approximately 30 minutes and was		

	conducted on separate days. The clinical and neuropsychological evaluation was performed before the onset of the training (pre-training) and at the end of the training phase (post-training). He or she would be told about a 3-min task involving moving around, reading, and memorizing the items on a memo pad placed on screen. They then took out those items from the shop trays and refrigerators after a period of distraction. The time taken to finish the task was recorded. During the 3-month booster training phase, 2 VR session were administered every week. Each VR session lasted approximately 15 minutes and was followed, after a pause of 1 minute, by 15 minutes in which the participant was invited to make an oral summary of the experience. The therapist-led training adopted a psychoeducational approach and was very similar to the VR, but with traditional activities. Colour-print images that matched the VR images should be made if a greater similarity between conventional and VR therapy are expected.		
Baseline Neuro psychological Evaluation	General cognitive abilities Verbal Memory Executive functions Visuospatial processing Daily living activities Depression		Optale et al., 2010

	Gender, n (%) Male) Female Level of education, n (%) <1 year 1–2 years >2 years Mean age (SD), years Multifactorial Memory Questionnaire and Fuld Object Memory Evaluation. CDR scale. Mini Mental State Examination (MMSE) The MMSE was developed by Folstein's group (Folstein et al., 1975) Geriatric Depression Scale (Yesavage et al., 1983). Multifactorial Memory Questionnaire (MMQ; Troyer and Rich, 2002). Fuld Object Memory Evaluation (FOME; Fuld, 1977) Lawton Instrumental Activities of Daily Living (HKLawton IADL) scale (Lawton and Brody, 1969; Tong and Man, 2002).		Man et al., 2012
Levels of difficulty	**Home setting** The home setting consisted of two bedrooms, one living room, one dining room, a kitchen, and a		Man et al., 2012

	bathroom. For instance, during training, each participant would be given instructions by both verbal and written messages displayed by the computer. He or she would be told about a 3-min task involving moving around, reading, and memorizing the items on a memo pad placed on the table within the living room. They then took out those items from the refrigerator in the kitchen after a period of distraction. The time taken to finish the task was recorded. **Convenience shop** The shop scenario was a simulated convenience shop consisting of six goods trays, one fruit tray, six refrigerators, and one cashier. Participants were asked to search around the shop and buy the requested items. Similar to the home scenario. In a second phase "Home version" they will use the joystick to control the "walk around" within the virtual shop to find and pick the correct items. The possibility of simulating the payment at the end of the tour seems interesting.		
Collected Data	Speed of walk (duration of distraction, for the Joystick version) Number of objects memorised (1-7)		

	Related objects: (by categories: fridge, office, etc. by functionality: to cook something, to have a bath, etc.) **Patient and therapist interface:** Patient privileges: Generic data Session number Selection between 3 levels of difficulty depending on: Speed, Number of objects, categories and functionality. Therapist privileges: **INFORMATION** History sessions by patient Number and type of errors Walk duration		
Expected results	VR group showing greater improvement in objective memory performance and the non-VR group showing better subjective memory subtest results in the Multifactorial Memory Questionnaire. When comparing the differential benefits of the two memory training programmes (VR versus non-VR), the findings suggest that the VR training programme led to significantly more gains in objective memory performance, in particular immediate recall and delayed recall of episodic memory. The use of multiple sensory modalities in a virtual environment and better focus of attention (Munro et al., 2002) may partly explain the improvement. Previous studies also suggested that		Man et al., 2012

	the VR training medium reduced the cognitive load by eliminating the need to convert two-dimensional training materials into three-dimensional representation that enables users to focus more cognitive resources on learning the task (Johnson and Hyde, 1997). Usability and motivational factors (Priore et al., 2003), better sense of control over memory abilities (Lachman, 2000), real-time interaction with the system (Bird and Kinsella, 1996), and feedback received on own performance (Riva, 2002) may also contribute to the success of the VR memory training in objective memory functioning. This study has some inherent limitations requiring attention. One of the limitations is the small number of participants. Also, there was no follow-up analysis. The training was considered short in duration and low in frequency (Rapp et al., 2002; Talassi et al., 2007)		

Annex B. Audios

TUTORIAL

- Bienvenido a Vr Supermarket EXperience. Con el fin de mejorar la adaptación a la realidad virtual a continuación empezaremos el tutorial. A partir de ahora todas las instrucciones del tutorial podrán volver a reproducirse si usted lo necesita pulsando el botón del lateral del mando, en la parte inferior.

- En esta primera parte vamos a aprender a girar sobre nosotros mismos para observar nuestro alrededor. El primer paso será mirar la esfera roja que tiene enfrente hasta que cambie a color verde.

- Genial. Así es cómo deberá actuar para desplazarse hacia delante. Si quiere desplazarse hacia la izquierda deberá girar sobre si mismo hacia esa dirección. Inténtelo

- ¡Muy bien! Ahora dese la vuelta para mirar a la esfera restante

- ¡Enhorabuena! Ya estamos finalizando la primera parte del tutorial. Ahora procederemos a explicarle como desplazarse usando los mandos. Mire a la puerta que se encuentra a su derecha. Apriete el panel central del mando derecho en la dirección en la que desea moverse. Por ejemplo si se quiere mover hacia la derecha deberá presionar la parte derecha del panel central. Si se quiere mover hacia la izquierda deberá presionar la parte izquierda. Si quiere avanzar hacia delante deberá presionar la parte superior del botón central. Si por lo contrario quiere retroceder deberá presionar la parte inferior. Siguiendo estos pasos ya se puede acercar a la puerta para seguir con el tutorial.

- ¡Ahora pondremos en práctica lo aprendido! Siga las flechas para llegar al final del pasillo

- ¡Perfecto! Ya nos queda poco. Ahora vamos a familiarizarnos con el ambiente del programa y vamos a interactuar con los objetos que nos encontremos. Deberá coger el paquete de castañas que se encuentra en la parte izquierda de la mesa central y colocarlo en la silueta que aparece en el estante de enfrente. Para coger los objetos debe presionar el gatillo trasero del mando derecho.

- Ahora fíjese que ha aparecido un objeto nuevo en la mesa. Deberá colocarlo en la silueta que aparece en la estantería de enfrente.

- Por último, como ha hecho con los elementos anteriores, deberá colocar el nuevo objeto que ha aparecido en su correspondiente lugar.

- ¡Felicidades! Ha finalizado con éxito el tutorial

TEST SCENE

- Ahora se podrá familiarizar con el entorno en el que trabajará más adelante. Se puede mover por todo el supermercado tal y como ha aprendido en el tutorial. También puede coger los objetos que desee. Para poner los objetos en la cesta que tiene situada en el mando izquierdo deberá acercar el objeto a la cesta y automáticamente se colocará dentro de la cesta. Cuando desee salir del tutorial tendrá que ir a la puerta de salida del supermercado. ¡Adelante!

SHOPPING LIST

- Hoy hay que hacer la compra. En la lista aparecen los productos que deberá poner en la cesta. La lista desaparecerá al cabo de unos segundos. Cuando no recuerde los productos que hay en la lista puede pulsar el botón del menú del mando izquierdo para que le vuelva a aparecer. ¡Comencemos!

- ¡Buenos días! Esperamos que tenga una buena compra.

- ¡Hola buenas! ¿Se ha fijado usted en las ofertas de la sección de congelados?

- ¡Enhorabuena! Ha completado con éxito el ejercicio. Ahora ya se puede dirigir a la puerta de salida del supermercado para terminar el nivel.

REPONEDOR

- En el centro del supermercado hay una mesa con diferentes productos. Deberá colocar los productos de la mesa en el estante que le corresponda del supermercado. ¡Adelante!

- ¡Enhorabuena! Ha completado con éxito el ejercicio.

COMUNIDADADES

- En la habitación hay estantes con diferentes banderas de comunidades autónomas. En la otra parte de la habitación se encuentra una mesa con diferentes productos. Cada producto es típico de una comunidad autónoma distinta. Tendrá que colocar cada producto en el estante correspondiente. ¡Comencemos!

- ¡Enhorabuena! Ha completado con éxito el ejercicio.

Annex C. Program Codes

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

public class accion : MonoBehaviour
{
    AudioSource audiosourcerepetir1;
    public AudioClip audiorepetir1;
    AudioSource audiosourcerepetir2;
    public AudioClip audiorepetir2;
    AudioSource audiosourcerepetir3;
    public AudioClip audiorepetir3;
    AudioSource audiosourcerepetir4;
    public AudioClip audiorepetir4;
    AudioSource audiosourcerepetir5;
    public AudioClip audiorepetir5;
    AudioSource audiosourcerepetir6;
    public AudioClip audiorepetir6;
    private SteamVR_TrackedController controller;
    public GameObject obj1;
    public GameObject obj2;
    public GameObject obj3;
    public GameObject marca1;
    public GameObject marca2;
    public GameObject marca3;
    public GameObject marca4;
    AudioSource activar;
    RaycastTuto codigo1;
    CloseInitialDoors codigo2;
    CloseFinalDoors codigo3;
    AudioSource variable;
    private int cont0 = 0;
    private int cont1 = 0;
    private int cont2 = 0;
    private int cont3 = 0;
    private int cont4 = 0;

    // Use this for initialization
    void Start()
    {
        StartCoroutine(voz());
        StartCoroutine(flag1());
        StartCoroutine(flag2());
        StartCoroutine(flag3());
    }

    private IEnumerator flag3()
    {
        yield return new WaitForSeconds(1.0f);
        codigo3 = marca3.GetComponent<CloseFinalDoors>();
    }
```

```
    private IEnumerator flag2()
    {
        yield return new WaitForSeconds(1.0f);
        codigo2 = marca2.GetComponent<CloseInitialDoors>();
    }

    private IEnumerator flag1()
    {
        yield return new WaitForSeconds(1.0f);
        codigo1 = marca1.GetComponent<RaycastTuto>();
    }

    private IEnumerator voz()
    {
        yield return new WaitForSeconds(33.0f);
        controller = GetComponentInParent<SteamVR_TrackedController>();
        controller.Gripped += repeticion;
    }

    private void repeticion(object sender, ClickedEventArgs e)
    {
        variable.Play();
    }

    void Update()
    {
        if (obj1.GetComponent<Renderer>().material.color == Color.green)
        {

            if (obj2.GetComponent<Renderer>().material.color ==
Color.green)
            {

                if (obj3.GetComponent<Renderer>().material.color ==
Color.green)
                {

                    if (codigo2.contador1 == 1)
                    {

                        if (cont4 == 0)
                        {
                            audiosourcerepetir4.clip = null;
                            audiosourcerepetir4.volume = 0;
                            StartCoroutine(pasillo());
                        }

                    }

                    else
                    {
                        if (cont3 == 0)
                        {
                            audiosourcerepetir3.clip = null;
                            audiosourcerepetir3.volume = 0;
                            StartCoroutine(esperargiro());
                        }
```

```
                }
            }

            else
            {
                if (cont2 == 0)
                {
                    audiosourcerepetir2.clip = null;
                    audiosourcerepetir2.volume = 0;
                    StartCoroutine(esperarderech());
                }

            }
        }

        else
        {
            if (cont1 == 0)
            {
                activar = marca4.GetComponent<AudioSource>();
                activar.clip = null;
                StartCoroutine(esperarizq());
            }

        }
    }

    else
    {
        if (cont0 == 0)
        {
            audiosourcerepetir1 = GetComponent<AudioSource>();
            audiosourcerepetir1.clip = audiorepetir1;
            variable = audiosourcerepetir1;
            cont0 = 1;
        }

    }

}

private IEnumerator pasillo()
{
    yield return new WaitForSeconds(codigo2.direccion.length);
    audiosourcerepetir5 = GetComponent<AudioSource>();
    audiosourcerepetir5.volume = 1;
    audiosourcerepetir5.clip = audiorepetir5;
    variable = audiosourcerepetir5;
    cont4 = 1;
}

private IEnumerator esperarizq()
{
    yield return new WaitForSeconds(codigo1.audioizquierda.length);
    audiosourcerepetir2 = GetComponent<AudioSource>();
    audiosourcerepetir2.volume = 1;
    audiosourcerepetir2.clip = audiorepetir2;
```

```
        variable = audiosourcerepetir2;
        cont1 = 1;
    }

    private IEnumerator esperarderech()
    {
        yield return new WaitForSeconds(codigo1.audioderecha.length);
        audiosourcerepetir3 = GetComponent<AudioSource>();
        audiosourcerepetir3.volume = 1;
        audiosourcerepetir3.clip = audiorepetir3;
        variable = audiosourcerepetir3;
        cont2 = 1;
    }

    private IEnumerator esperargiro()
    {
        yield return new WaitForSeconds(codigo1.audiogiro.length);
        audiosourcerepetir4 = GetComponent<AudioSource>();
        audiosourcerepetir4.volume = 1;
        audiosourcerepetir4.clip = audiorepetir4;
        variable = audiosourcerepetir4;
        cont3 = 1;
    }
}
```

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

public class activar : MonoBehaviour {

    private SteamVR_TrackedController controller;
    public GameObject cubo;
    private Collider espacio;

    void Start()
    {
        controller = GetComponentInParent<SteamVR_TrackedController>();
    }

    void Update()
    {
        if (controller.triggerPressed == true)
        {
            espacio = cubo.GetComponent<Collider>();
            espacio.enabled = true;
        }
        else
        {
            espacio = cubo.GetComponent<Collider>();
            espacio.enabled = false;
        }
    }
}
```

```
using System;
using UnityEngine;
using Object = UnityEngine.Object;

namespace UnityStandardAssets.Utility
{
    public class ActivateTrigger : MonoBehaviour
    {
        // A multi-purpose script which causes an action to occur when
        // a trigger collider is entered.
        public enum Mode
        {
            Trigger = 0,    // Just broadcast the action on to the target
            Replace = 1,    // replace target with source
            Activate = 2,   // Activate the target GameObject
            Enable = 3,     // Enable a component
            Animate = 4,    // Start animation on target
            Deactivate = 5  // Decativate target GameObject
        }

        public Mode action = Mode.Activate;         // The action to
accomplish
        public Object target;                       // The game object to
affect. If none, the trigger work on this game object
        public GameObject source;
        public int triggerCount = 1;
        public bool repeatTrigger = false;


        private void DoActivateTrigger()
        {
            triggerCount--;

            if (triggerCount == 0 || repeatTrigger)
            {
                Object currentTarget = target ?? gameObject;
                Behaviour targetBehaviour = currentTarget as Behaviour;
                GameObject targetGameObject = currentTarget as
GameObject;
                if (targetBehaviour != null)
                {
                    targetGameObject = targetBehaviour.gameObject;
                }

                switch (action)
                {
                    case Mode.Trigger:
                        if (targetGameObject != null)
                        {

targetGameObject.BroadcastMessage("DoActivateTrigger");
                        }
                        break;
                    case Mode.Replace:
                        if (source != null)
                        {
                            if (targetGameObject != null)
                            {
```

```
                                Instantiate(source,
targetGameObject.transform.position,

targetGameObject.transform.rotation);
                                DestroyObject(targetGameObject);
                            }
                        }
                        break;
                    case Mode.Activate:
                        if (targetGameObject != null)
                        {
                            targetGameObject.SetActive(true);
                        }
                        break;
                    case Mode.Enable:
                        if (targetBehaviour != null)
                        {
                            targetBehaviour.enabled = true;
                        }
                        break;
                    case Mode.Animate:
                        if (targetGameObject != null)
                        {

targetGameObject.GetComponent<Animation>().Play();
                        }
                        break;
                    case Mode.Deactivate:
                        if (targetGameObject != null)
                        {
                            targetGameObject.SetActive(false);
                        }
                        break;
                }
            }
        }


        private void OnTriggerEnter(Collider other)
        {
            DoActivateTrigger();
        }
    }
}
```

```
using UnityEngine;
using MySql.Data.MySqlClient;
using System.Data;

public class AdminMYSQL : MonoBehaviour {
    public string servidorBaseDatos;
    public string nombreBaseDatos;
    public string usuarioBaseDatos;
    public string contraseñaBaseDatos;

    private string datosConexion;
    private MySqlConnection conexion;


        // Use this for initialization
        void Start () {
        datosConexion = "Server=" + servidorBaseDatos
                        + ";Database=" + nombreBaseDatos
                        + ";Uid=" + usuarioBaseDatos
                        + ";Pwd=" + contraseñaBaseDatos
                        + ";";
        ConectarConServidorBaseDatos();

    }

    private void ConectarConServidorBaseDatos()
    {
        conexion = new MySqlConnection(datosConexion);
        try
        {
            conexion.Open();
            Debug.Log("Conexion con BD correcta");
        }
        catch (MySqlException error)
        {
            Debug.LogError("Imposible conectar con las Base de Datos" +
error);
        }
    }
        public MySqlDataReader Select(string _select)
        {
            MySqlCommand cmd = conexion.CreateCommand();
            cmd.CommandText = "SELECT * FROM " + _select;
            MySqlDataReader Resultado = cmd.ExecuteReader();
        return Resultado;
        }

        public MySqlDataReader Insert(string _insert)
        {
        MySqlCommand cmd = conexion.CreateCommand();
        cmd.CommandText = "INSERT INTO " + _insert;
        MySqlDataReader Resultado = cmd.ExecuteReader();
        return Resultado;
        }

}
```

```
namespace VRTK.Examples.Archery
{
    using UnityEngine;

    public class Arrow : MonoBehaviour
    {
        public float maxArrowLife = 10f;
        [HideInInspector]
        public bool inFlight = false;

        private bool collided = false;
        private Rigidbody rigidBody;
        private GameObject arrowHolder;
        private Vector3 originalPosition;
        private Quaternion originalRotation;
        private Vector3 originalScale;

        public void SetArrowHolder(GameObject holder)
        {
            arrowHolder = holder;
            arrowHolder.SetActive(false);
        }

        public void OnNock()
        {
            collided = false;
            inFlight = false;
        }

        public void Fired()
        {
            DestroyArrow(maxArrowLife);
        }

        public void ResetArrow()
        {
            collided = true;
            inFlight = false;
            RecreateNotch();
            ResetTransform();
        }

        private void Start()
        {
            rigidBody = GetComponent<Rigidbody>();
            SetOrigns();
        }

        private void SetOrigns()
        {
            originalPosition = transform.localPosition;
            originalRotation = transform.localRotation;
            originalScale = transform.localScale;
        }

        private void FixedUpdate()
        {
```

```
            if (!collided)
            {
                transform.LookAt(transform.position +
rigidBody.velocity);
            }
        }

        private void OnCollisionEnter(Collision collision)
        {
            if (inFlight)
            {
                ResetArrow();
            }
        }

        private void RecreateNotch()
        {
            //swap the arrow holder to be the parent again
            arrowHolder.transform.SetParent(null);
            arrowHolder.SetActive(true);

            //make the arrow a child of the holder again
            transform.SetParent(arrowHolder.transform);

            //reset the state of the rigidbodies and colliders
            GetComponent<Rigidbody>().isKinematic = true;
            GetComponent<Collider>().enabled = false;
            arrowHolder.GetComponent<Rigidbody>().isKinematic = false;
        }

        private void ResetTransform()
        {
            arrowHolder.transform.position = transform.position;
            arrowHolder.transform.rotation = transform.rotation;
            transform.localPosition = originalPosition;
            transform.localRotation = originalRotation;
            transform.localScale = originalScale;
        }

        private void DestroyArrow(float time)
        {
            Destroy(arrowHolder, time);
            Destroy(gameObject, time);
        }
    }
}
```

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

public class audio_inicio_reponedor : MonoBehaviour
{
    public GameObject LeftController;
    public GameObject RightController;
    AudioSource audiosourceinicio;
    public AudioClip AudioInicio;
    // Use this for initialization
    void Start()
    {
        audiosourceinicio = GetComponent<AudioSource>();
        audiosourceinicio.clip = AudioInicio;
        audiosourceinicio.Play();
        LeftController.SetActive(false);
        RightController.SetActive(false);
        StartCoroutine(tiempoaudioinicial());
    }

    IEnumerator tiempoaudioinicial()
    {
        yield return new WaitForSeconds(AudioInicio.length);
        LeftController.SetActive(true);
        RightController.SetActive(true);
    }
}
```

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

public class AudioInicioMainScene : MonoBehaviour {
    public GameObject LeftController;
    public GameObject RightController;
    public GameObject Nota;
    public GameObject Timer;
    public GameObject ProductosCesta;
    public GameObject Objetos;


    AudioSource audiosourceinicio;

    public AudioClip AudioInicio;


    // Use this for initialization
    void Start()
    {
        audiosourceinicio = GetComponent<AudioSource>();
        audiosourceinicio.clip = AudioInicio;
        audiosourceinicio.Play();
        LeftController.SetActive(false);
        RightController.SetActive(false);
        StartCoroutine(tiempoaudioinicial());

    }
    IEnumerator tiempoaudioinicial()
    {
        yield return new WaitForSeconds(AudioInicio.length);
        Nota.SetActive(true);
        LeftController.SetActive(true);
        RightController.SetActive(true);
        ProductosCesta.SetActive(true);
        Objetos.SetActive(true);

    }
}
```

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

public class AudioInicioMainScene23 : MonoBehaviour {
    public GameObject LeftController;
    public GameObject RightController;
    AudioSource audiosourceinicio;
    public AudioClip AudioInicio;
    public GameObject Nota;
    public GameObject Imagen;
    public GameObject Timer;
    public GameObject ProductosCesta;
    public GameObject Objetos;

    // Use this for initialization
    void Start () {
        audiosourceinicio = GetComponent<AudioSource>();
        audiosourceinicio.clip = AudioInicio;
        audiosourceinicio.Play();
        LeftController.SetActive(false);
        RightController.SetActive(false);
        StartCoroutine(tiempoaudioinicial());
        StartCoroutine(imagenmando());

    }

    private IEnumerator imagenmando()
    {
        yield return new WaitForSeconds(10.0f);
        Imagen.SetActive(true);
    }

    IEnumerator tiempoaudioinicial()
    {
        yield return new WaitForSeconds(AudioInicio.length);
        Imagen.SetActive(false);
        Nota.SetActive(true);
        StartCoroutine(iniciolista());


    }

    private IEnumerator iniciolista()
    {
        yield return new WaitForSeconds(10.0f);
        ProductosCesta.SetActive(true);
        Objetos.SetActive(true);
        Nota.SetActive(false);
        LeftController.SetActive(true);
        RightController.SetActive(true);
    }
}
```

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

public class AudioInicioTuto : MonoBehaviour
{
    public GameObject LeftController;
    public GameObject RightController;
    public GameObject ProductosCesta;
    public GameObject Objetos;
    public GameObject trigger;
    AudioSource audiosourceinicio;
    public AudioClip AudioInicio;


    // Use this for initialization
    void Start()
    {
        audiosourceinicio = GetComponent<AudioSource>();
        audiosourceinicio.clip = AudioInicio;
        audiosourceinicio.Play();
        LeftController.SetActive(false);
        RightController.SetActive(false);
        StartCoroutine(tiempoaudioinicial());
        StartCoroutine(inivideo());

    }

    private IEnumerator inivideo()
    {
        yield return new WaitForSeconds(11.0f);
        trigger.SetActive(true);
        StartCoroutine(finvideo());
    }

    private IEnumerator finvideo()
    {
        yield return new WaitForSeconds(12.0f);
        trigger.SetActive(false);
    }

    IEnumerator tiempoaudioinicial()
    {
        yield return new WaitForSeconds(AudioInicio.length);
        LeftController.SetActive(true);
        RightController.SetActive(true);
        ProductosCesta.SetActive(true);
        Objetos.SetActive(true);

    }
}
```

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

public class BotonRepetir : MonoBehaviour {
    public


    void Awake()
    {
        // Call the LevelManager and set the last level.
        //SceneManager.setLastLevel(Application.loadedLevelName);
        LevelManager.getLastLevel();
    }

}
```

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


public class ChangeComunidades1 : MonoBehaviour
{


    public int id;



    ProductosCestaController productosCestaController;

    void Awake()
    {

VRTK_SDKManager.instance.AddBehaviourToToggleOnLoadedSetupChange(this);
    }

    private void Start()
    {
        productosCestaController =
FindObjectOfType<ProductosCestaController>();
    }

    void OnDestroy()
    {

VRTK_SDKManager.instance.RemoveBehaviourToToggleOnLoadedSetupChange(this)
;
    }
    void OnEnable()
    {

        GetComponent<VRTK_SnapDropZone>().ObjectSnappedToDropZone +=
ChangeLevel_ObjectSnappedToDropZone;

        object test1 = null;
        SnapDropZoneEventArgs test2 = new SnapDropZoneEventArgs();
        ChangeLevel_ObjectSnappedToDropZone(test1, test2);

    }
    void OnDisable()
    {

        GetComponent<VRTK_SnapDropZone>().ObjectSnappedToDropZone -=
ChangeLevel_ObjectSnappedToDropZone;
    }


    private void ChangeLevel_ObjectSnappedToDropZone(object sender,
SnapDropZoneEventArgs e)
    {
```

```
        if (e.snappedObject.name == "fuet")
        {
            productosCestaController.SetObjectTrue(id);

        }

        if (e.snappedObject.name == "platano")
        {
            productosCestaController.SetObjectTrue(id);

        }

        if (e.snappedObject.name == "gazpacho")
        {
            productosCestaController.SetObjectTrue(id);

        }
    }
}
```

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


public class ChangeComunidades2 : MonoBehaviour
{


    public int id;



    ProductosCestaController productosCestaController;

    void Awake()
    {

VRTK_SDKManager.instance.AddBehaviourToToggleOnLoadedSetupChange(this);
    }

    private void Start()
    {
        productosCestaController =
FindObjectOfType<ProductosCestaController>();
    }

    void OnDestroy()
    {

VRTK_SDKManager.instance.RemoveBehaviourToToggleOnLoadedSetupChange(this)
;
    }
    void OnEnable()
    {

        GetComponent<VRTK_SnapDropZone>().ObjectSnappedToDropZone +=
ChangeLevel_ObjectSnappedToDropZone;

        object test1 = null;
        SnapDropZoneEventArgs test2 = new SnapDropZoneEventArgs();
        ChangeLevel_ObjectSnappedToDropZone(test1, test2);

    }
    void OnDisable()
    {

        GetComponent<VRTK_SnapDropZone>().ObjectSnappedToDropZone -=
ChangeLevel_ObjectSnappedToDropZone;
    }


    private void ChangeLevel_ObjectSnappedToDropZone(object sender,
SnapDropZoneEventArgs e)
    {
```

```
        if (e.snappedObject.name == "mojopicon")
        {
            productosCestaController.SetObjectTrue(id);

        }

        if (e.snappedObject.name == "aceitunas")
        {
            productosCestaController.SetObjectTrue(id);

        }

        if (e.snappedObject.name == "sobrasada")
        {
            productosCestaController.SetObjectTrue(id);

        }
        if (e.snappedObject.name == "fabada")
        {
            productosCestaController.SetObjectTrue(id);

        }
        if (e.snappedObject.name == "cava")
        {
            productosCestaController.SetObjectTrue(id);

        }
    }
}
```

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


public class ChangeComunidades3 : MonoBehaviour
{


    public int id;



    ProductosCestaController productosCestaController;

    void Awake()
    {

VRTK_SDKManager.instance.AddBehaviourToToggleOnLoadedSetupChange(this);
    }

    private void Start()
    {
        productosCestaController =
FindObjectOfType<ProductosCestaController>();
    }

    void OnDestroy()
    {

VRTK_SDKManager.instance.RemoveBehaviourToToggleOnLoadedSetupChange(this)
;
    }
    void OnEnable()
    {

        GetComponent<VRTK_SnapDropZone>().ObjectSnappedToDropZone +=
ChangeLevel_ObjectSnappedToDropZone;

        object test1 = null;
        SnapDropZoneEventArgs test2 = new SnapDropZoneEventArgs();
        ChangeLevel_ObjectSnappedToDropZone(test1, test2);

    }
    void OnDisable()
    {

        GetComponent<VRTK_SnapDropZone>().ObjectSnappedToDropZone -=
ChangeLevel_ObjectSnappedToDropZone;
    }


    private void ChangeLevel_ObjectSnappedToDropZone(object sender,
SnapDropZoneEventArgs e)
    {
```

```
        if (e.snappedObject.name == "txakoli")
        {
            productosCestaController.SetObjectTrue(id);

        }

        if (e.snappedObject.name == "vinotinto")
        {
            productosCestaController.SetObjectTrue(id);

        }

        if (e.snappedObject.name == "mejillones")
        {
            productosCestaController.SetObjectTrue(id);

        }
        if (e.snappedObject.name == "naranjas")
        {
            productosCestaController.SetObjectTrue(id);

        }
        if (e.snappedObject.name == "cerezas")
        {
            productosCestaController.SetObjectTrue(id);

        }
        if (e.snappedObject.name == "ensaimada")
        {
            productosCestaController.SetObjectTrue(id);

        }
        if (e.snappedObject.name == "morcilla")
        {
            productosCestaController.SetObjectTrue(id);

        }
    }
}
```

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


public class ChangeLevel1 : MonoBehaviour
{


    public int id;
    private int _leche;
    private int _pizza;
    private int _bimbo;



    ProductosCestaController productosCestaController;

    void Awake()
    {

VRTK_SDKManager.instance.AddBehaviourToToggleOnLoadedSetupChange(this);
    }

    private void Start()
    {
        productosCestaController =
FindObjectOfType<ProductosCestaController>();
    }

    void OnDestroy()
    {

VRTK_SDKManager.instance.RemoveBehaviourToToggleOnLoadedSetupChange(this)
;
    }
    void OnEnable()
    {

        GetComponent<VRTK_SnapDropZone>().ObjectSnappedToDropZone +=
ChangeLevel_ObjectSnappedToDropZone;

        object test1 = null;
        SnapDropZoneEventArgs test2 = new SnapDropZoneEventArgs();
        ChangeLevel_ObjectSnappedToDropZone(test1, test2);

    }
    void OnDisable()
    {

        GetComponent<VRTK_SnapDropZone>().ObjectSnappedToDropZone -=
ChangeLevel_ObjectSnappedToDropZone;
    }
```

```
    private void ChangeLevel_ObjectSnappedToDropZone(object sender,
SnapDropZoneEventArgs e)
    {


        if (e.snappedObject.name == "leche")
        {
            productosCestaController.SetObjectTrue(id);


        }

        if (e.snappedObject.name == "bimbo")
        {
            productosCestaController.SetObjectTrue(id);

        }

        if (e.snappedObject.name == "pizza")
        {
            productosCestaController.SetObjectTrue(id);

        }
    }
}
```

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


public class ChangeLevel2 : MonoBehaviour
{


    public int id;



    ProductosCestaController productosCestaController;

    void Awake()
    {

VRTK_SDKManager.instance.AddBehaviourToToggleOnLoadedSetupChange(this);
    }

    private void Start()
    {
        productosCestaController =
FindObjectOfType<ProductosCestaController>();
    }

    void OnDestroy()
    {

VRTK_SDKManager.instance.RemoveBehaviourToToggleOnLoadedSetupChange(this)
;
    }
    void OnEnable()
    {

        GetComponent<VRTK_SnapDropZone>().ObjectSnappedToDropZone +=
ChangeLevel_ObjectSnappedToDropZone;

        object test1 = null;
        SnapDropZoneEventArgs test2 = new SnapDropZoneEventArgs();
        ChangeLevel_ObjectSnappedToDropZone(test1, test2);

    }
    void OnDisable()
    {

        GetComponent<VRTK_SnapDropZone>().ObjectSnappedToDropZone -=
ChangeLevel_ObjectSnappedToDropZone;
    }


    private void ChangeLevel_ObjectSnappedToDropZone(object sender,
SnapDropZoneEventArgs e)
    {
```

```
        if (e.snappedObject.name == "harina")
        {
            productosCestaController.SetObjectTrue(id);

        }

        if (e.snappedObject.name == "berenjena")
        {
            productosCestaController.SetObjectTrue(id);

        }

        if (e.snappedObject.name == "piña")
        {
            productosCestaController.SetObjectTrue(id);
        }

        if (e.snappedObject.name == "ternera")
        {
            productosCestaController.SetObjectTrue(id);
        }


    }

}
```

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


public class ChangeLevel3 : MonoBehaviour
{


    public int id;



    ProductosCestaController productosCestaController;

    void Awake()
    {

VRTK_SDKManager.instance.AddBehaviourToToggleOnLoadedSetupChange(this);
    }

    private void Start()
    {
        productosCestaController =
FindObjectOfType<ProductosCestaController>();
    }

    void OnDestroy()
    {

VRTK_SDKManager.instance.RemoveBehaviourToToggleOnLoadedSetupChange(this)
;
    }
    void OnEnable()
    {

        GetComponent<VRTK_SnapDropZone>().ObjectSnappedToDropZone +=
ChangeLevel_ObjectSnappedToDropZone;

        object test1 = null;
        SnapDropZoneEventArgs test2 = new SnapDropZoneEventArgs();
        ChangeLevel_ObjectSnappedToDropZone(test1, test2);

    }
    void OnDisable()
    {

        GetComponent<VRTK_SnapDropZone>().ObjectSnappedToDropZone -=
ChangeLevel_ObjectSnappedToDropZone;
    }


    private void ChangeLevel_ObjectSnappedToDropZone(object sender,
SnapDropZoneEventArgs e)
    {
```

```
        if (e.snappedObject.name == "Chips")
        {
            productosCestaController.SetObjectTrue(id);

        }

        if (e.snappedObject.name == "Endivia")
        {
            productosCestaController.SetObjectTrue(id);

        }

        if (e.snappedObject.name == "Barra Pan")
        {
            productosCestaController.SetObjectTrue(id);

        }
        if (e.snappedObject.name == "Guisantes")
        {
            productosCestaController.SetObjectTrue(id);

        }

        if (e.snappedObject.name == "pasta")
        {
            productosCestaController.SetObjectTrue(id);

        }

        if (e.snappedObject.name == "Yogurt")
        {
            productosCestaController.SetObjectTrue(id);

        }

        if (e.snappedObject.name == "sandia")
        {
            productosCestaController.SetObjectTrue(id);

        }
    }


}
```

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

public class CloseFinalDoors : MonoBehaviour {
        public GameObject door1;
        public GameObject door2;
        public GameObject snap1;
        public GameObject controllerLeft;
        public GameObject controllerRight;
    public GameObject PlanoTrigger;
    Animator animation1;
        Animator animation2;
        AudioSource audiosourcepuerta;
        AudioSource audiosourceexplicacion;
        //public AudioClip puerta;
        public AudioClip explicacion;
        public int contador2;

        // Use this for initialization
        void Start () {
                animation1 = door1.GetComponent<Animator>();
                animation2 = door2.GetComponent<Animator>();
                audiosourcepuerta = door1.GetComponent<AudioSource> ();
                //audiosourcepuerta.clip = puerta;
                audiosourceexplicacion = snap1.GetComponent<AudioSource>
();
                audiosourceexplicacion.clip = explicacion;
        }
                void OnTriggerEnter (Collider collider)
        {
                animation1.enabled= true;
                animation2.enabled= true;
                contador2 = contador2 + 1;
                if (contador2 == 1) {

                        //audiosourcepuerta.Play();
                        audiosourceexplicacion.Play ();
                        controllerLeft.SetActive (false);
                        controllerRight.SetActive (false);
                        StartCoroutine (tiempo ());
            StartCoroutine (planoTrigger());
         }

        }
        IEnumerator tiempo()
        {
                yield return new WaitForSeconds (explicacion.length);
                controllerLeft.SetActive (true);
                controllerRight.SetActive (true);
         PlanoTrigger.SetActive(false);
    }
    IEnumerator planoTrigger()
    {
         yield return new WaitForSeconds(18.0f);
         PlanoTrigger.SetActive(true);
    }
}
```

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

public class CloseInitialDoors : MonoBehaviour {
        public GameObject door1;
        public GameObject door2;
        public GameObject arrow1;
        Animator animation1;
        Animator animation2;
        public int contador1 = 0;
    public int contador2 = 0;
    AudioSource audiosourcedireccion;
        //public AudioClip puerta;
        public AudioClip direccion;
        // Use this for initialization
        void Start () {
                animation1 = door1.GetComponent<Animator>();
                animation2 = door2.GetComponent<Animator>();
                audiosourcedireccion = arrow1.GetComponent<AudioSource>
();
                audiosourcedireccion.clip = direccion;
        }


        void OnTriggerEnter (Collider collider)
        {
                animation1.SetBool ("close",true);
                animation2.SetBool ("close",true);
                contador1 = 1;
                if ((contador1 == 1) & (contador2 == 0)) {
                        audiosourcedireccion.Play ();
            contador2 = 1;
                }

        }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;
 using MySql.Data.MySqlClient;

public class Contador : MonoBehaviour {


    private float contador = 5.0f;
    int aciertos = 0;
    string escena;
    string levelname;


    public void Aciertos(int valor)
    {
        aciertos = valor;
    }


}
```

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

public class ContadorListaConsulta : MonoBehaviour {

        //public GameObject LeftController;
        public GameObject RightController;
        public GameObject Nota;
        public GameObject ListaCesta;
    public GameObject NombreProductosCesta;
    public int TiempoVista;

    public int ContadorConsulta=0;

        // Use this for initialization
        void Start () {

        }

        // Update is called once per frame
        public void Contador () {
                //LeftController.SetActive (false);
                RightController.SetActive (false);
                Nota.SetActive (true);
                ContadorConsulta = ContadorConsulta + 1;
                ListaCesta.SetActive (false);
         NombreProductosCesta.SetActive(false);
         StartCoroutine(TiempoMandos());
    }
        IEnumerator TiempoMandos()
        {
                yield return new WaitForSeconds(TiempoVista);
                //LeftController.SetActive (true);
                RightController.SetActive (true);
                Nota.SetActive (false);
                ListaCesta.SetActive (true);
         NombreProductosCesta.SetActive(true);
    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using Mono.Data.Sqlite;
using System.Data;
using System;
using UnityEngine.UI;

public class db : MonoBehaviour {
        public GameObject GuardarDatos;
        public GameObject Input;

        private int ID ;
        private int OV3 ;
        private int ONV3 ;
        private int T3 ;
        private int OV5 ;
        private int ONV5 ;
        private int T5 ;
        private int C5 ;
        private int OV7 ;
        private int ONV7 ;
        private int T7 ;
        private int C7 ;
    private int TR1;
    private int TR2;
    private int TR3;
    private int TC1;
    private int TC2;
    private int TC3;


    void Start () {
                //sqlite_BaseDatos ();
        }


        public void sqlite_BaseDatos ()

        {
                string conn = "URI=file:" + Application.dataPath +
"/plugins/BaseDatos.db"; //Path to database.
                IDbConnection dbconn;
                dbconn = (IDbConnection) new SqliteConnection(conn);
                dbconn.Open(); //Open connection to the database.

                IDbCommand dbcmd = dbconn.CreateCommand();
                string sqlQuery = "SELECT * FROM Datos WHERE ID=
"+Input.GetComponent<Text>().text;
                dbcmd.CommandText = sqlQuery;
                IDataReader reader = dbcmd.ExecuteReader();

                while (reader.Read())
                {
                        int ID = reader.GetInt32(0);
                        int OV3 = reader.GetInt32(1);
                        int ONV3 = reader.GetInt32(2);
                        int T3 = reader.GetInt32(3);
```

```
                        int OV5 = reader.GetInt32(4);
                        int ONV5 = reader.GetInt32(5);
                        int T5 = reader.GetInt32(6);
                        int C5 = reader.GetInt32(7);
                        int OV7 = reader.GetInt32(8);
                        int ONV7 = reader.GetInt32(9);
                        int T7 = reader.GetInt32(10);
                        int C7 = reader.GetInt32(11);
            int TR1 = reader.GetInt32(12);
            int TR2 = reader.GetInt32(13);
            int TR3 = reader.GetInt32(14);
            int TC1 = reader.GetInt32(15);
            int TC2 = reader.GetInt32(16);
            int TC3 = reader.GetInt32(17);


            Debug.Log( "ID= "+ID+"  Objetos válidos ="+OV3+" Objetos no
válidos ="+ ONV3+"Tiempo ="+
                                T3+"Objetos válidos 5="+OV5+"+ Objetos no
válidos5="+ONV5+" Tiempo="+T5+"Consutas 5="+C5+
                                "Objetos válidos 7="+OV7+"Objetos no
válidos 7="+ONV7+"Tiempo 7="+T7+"Consultas 7="+C7+"Tiempo R1="+TR1+
                "Tiempo R2=" + TR2 + "Tiempo R3=" + TR3 + "Tiempo C1=" +
TC1 + "Tiempo C2=" + TC2 + "Tiempo C3=" + TC3);


                        GuardarDatos.GetComponent<GameControl> ().ID = ID;
                        GuardarDatos.GetComponent<GameControl> ().OV3 =
OV3;
                        GuardarDatos.GetComponent<GameControl> ().ONV3 =
ONV3;
                        GuardarDatos.GetComponent<GameControl> ().T3 = T3;
                        GuardarDatos.GetComponent<GameControl> ().OV5 =
OV5;
                        GuardarDatos.GetComponent<GameControl> ().ONV5 =
ONV5;
                        GuardarDatos.GetComponent<GameControl> ().T5 = T5;
                        GuardarDatos.GetComponent<GameControl> ().C5 = C5;
                        GuardarDatos.GetComponent<GameControl> ().OV7 =
OV7;
                        GuardarDatos.GetComponent<GameControl> ().ONV7 =
ONV7;

                        GuardarDatos.GetComponent<GameControl> ().T7 = T7;
                        GuardarDatos.GetComponent<GameControl> ().C7 = C7;
            GuardarDatos.GetComponent<GameControl>().TR1 = TR1;
            GuardarDatos.GetComponent<GameControl>().TR2 = TR2;
            GuardarDatos.GetComponent<GameControl>().TR3 = TR3;
            GuardarDatos.GetComponent<GameControl>().TC1 = TC1;
            GuardarDatos.GetComponent<GameControl>().TC2 = TC2;
            GuardarDatos.GetComponent<GameControl>().TC3 = TC3;


        }
                reader.Close();
                reader = null;
```

```
                dbcmd.Dispose();
                dbcmd = null;

                dbconn.Close();
                dbconn = null;


        }
}
```

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

public class DesactivarNota : MonoBehaviour {


        // Use this for initialization
        void Start () {
                StartCoroutine (DesactivarNotas ());
        }

        IEnumerator DesactivarNotas()
        {
                yield return new WaitForSeconds (15);
                gameObject.SetActive (false);
        }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;
using Mono.Data.Sqlite;
using System.Data;
using System;
using UnityEngine.UI;

public class GameControl : MonoBehaviour {

        public static GameControl control;
        //public GameObject BaseDatos;
        private GameObject dentrocesta;
        private GameObject LeftController;
        private GameObject tiempo;
        public int ID;
        public int OV3 ;
        public int ONV3 ;
        public int T3 ;
        public int OV5 ;
        public int ONV5 ;
        public int T5 ;
        public int C5 ;
        public int OV7 ;
        public int ONV7 ;
        public int T7 ;
        public int C7 ;
    public int TR1;
    public int TR2;
    public int TR3;
    public int TC1;
    public int TC2;
    public int TC3;

    void Awake () {
                if (control == null) {
                        DontDestroyOnLoad (gameObject);
                        control = this;
                }
                else if(control !=this)
                {
                        Destroy(gameObject);
                }

        }



        public void GuardarDatos (){

                //ID = BaseDatos.GetComponent<db>().ID;
                //OV3 = BaseDatos.GetComponent<db> ().OV3;
                //ONV3 = BaseDatos.GetComponent<db> ().ONV3;
                //T3 = BaseDatos.GetComponent<db> ().T3;
                Debug.Log ("ID="+ ID + "OV3="+ OV3 + "ONV3="+ ONV3 +
"T3="+T3);
```

```
        }

    public void Escena1()
    {

        string conn = "URI=file:" + Application.dataPath +
"/plugins/BaseDatos.db"; //Path to database.
        IDbConnection dbconn;
        dbconn = (IDbConnection)new SqliteConnection(conn);
        dbconn.Open(); //Open connection to the database.

        IDbCommand dbcmd = dbconn.CreateCommand();
        string sqlQuery = "UPDATE Datos SET ObjetosVal3='" + OV3 + "'
WHERE ID=" + ID;
        dbcmd.CommandText = sqlQuery;
        IDataReader reader = dbcmd.ExecuteReader();


        reader.Close();
        reader = null;

        dbcmd.Dispose();
        dbcmd = null;

        dbconn.Close();
        dbconn = null;

        //dentrocesta = GameObject.Find ("DentroCesta");
        //OV3 = dentrocesta.GetComponent<StopGravity> ().Valid;
        //ONV3 = dentrocesta.GetComponent<StopGravity> ().NoVal;
        //T3 = dentrocesta.GetComponent<StopGravity> ().timeint;
        //Debug.Log ("ID="+ ID + "OV3="+ OV3 + "ONV3="+ ONV3 + "T3="+T3);
    }

    void Update()
    {


        if (SceneManager.GetActiveScene() ==
            SceneManager.GetSceneByName("Main scene"))
        {



            dentrocesta = GameObject.Find("DentroCesta");
            OV3 = dentrocesta.GetComponent<StopGravity>().Valid;
            ONV3 = dentrocesta.GetComponent<StopGravity>().NoVal;
            T3 = dentrocesta.GetComponent<StopGravity>().timeint;

        }
        if (SceneManager.GetActiveScene() ==
            SceneManager.GetSceneByName("Main sceneL2"))
        {


            tiempo = GameObject.Find("Script");
```

```
            T5 = tiempo.GetComponent<TimerRecogida>().timeint;
            LeftController = GameObject.Find("LeftController");
            C5 =
LeftController.GetComponent<ContadorListaConsulta>().ContadorConsulta;
            dentrocesta = GameObject.Find("DentroCesta");
            OV5 = dentrocesta.GetComponent<StopGravity5>().Valid;
            ONV5 = dentrocesta.GetComponent<StopGravity5>().NoVal;



        }
        if (SceneManager.GetActiveScene() ==
            SceneManager.GetSceneByName("Main sceneL3"))
        {

            tiempo = GameObject.Find("Script");
            T7 = tiempo.GetComponent<TimerRecogida>().timeint;
            LeftController = GameObject.Find("LeftController");
            C7 =
LeftController.GetComponent<ContadorListaConsulta>().ContadorConsulta;
            dentrocesta = GameObject.Find("DentroCesta");
            OV7 = dentrocesta.GetComponent<StopGravity7>().Valid;
            ONV7 = dentrocesta.GetComponent<StopGravity7>().NoVal;

        }

        if (SceneManager.GetActiveScene() ==
            SceneManager.GetSceneByName("reponedornivel1"))
        {

            tiempo = GameObject.Find("Script");
            TR1 = tiempo.GetComponent<TimerRecogida>().timeint;
            LeftController = GameObject.Find("LeftController");
        }

        if (SceneManager.GetActiveScene() ==
            SceneManager.GetSceneByName("reponedornivel2"))
        {

            tiempo = GameObject.Find("Script");
            TR2 = tiempo.GetComponent<TimerRecogida>().timeint;
            LeftController = GameObject.Find("LeftController");
        }

        if (SceneManager.GetActiveScene() ==
            SceneManager.GetSceneByName("reponedornivel3"))
        {

            tiempo = GameObject.Find("Script");
            TR3 = tiempo.GetComponent<TimerRecogida>().timeint;
            LeftController = GameObject.Find("LeftController");
        }

        if (SceneManager.GetActiveScene() ==
            SceneManager.GetSceneByName("comunidadesnivel1"))
        {

            tiempo = GameObject.Find("Script");
```

```
            TC1 = tiempo.GetComponent<TimerRecogida>().timeint;
            LeftController = GameObject.Find("LeftController");
        }

        if (SceneManager.GetActiveScene() ==
            SceneManager.GetSceneByName("comunidadesnivel2"))
        {

            tiempo = GameObject.Find("Script");
            TC2 = tiempo.GetComponent<TimerRecogida>().timeint;
            LeftController = GameObject.Find("LeftController");
        }

        if (SceneManager.GetActiveScene() ==
            SceneManager.GetSceneByName("comunidadesnivel3"))
        {

            tiempo = GameObject.Find("Script");
            TC3 = tiempo.GetComponent<TimerRecogida>().timeint;
            LeftController = GameObject.Find("LeftController");
        }

    }

}
```

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

public class GoToFinalTuto : MonoBehaviour {

        public AudioClip finalsound;

        // Use this for initialization
        void Start () {

                StartCoroutine (FinalTuto ());

        }
        IEnumerator FinalTuto()
        {
                yield return new WaitForSeconds (finalsound.length);
                SceneManager.LoadScene ("Test scene");
        }

}
```

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

public class InicioSaludo : MonoBehaviour {

        public GameObject Mujer;
        AudioSource AudioSourceMujer;
        public int contador = 0;
        //public AudioClip AudioSaludo;
        // Use this for initialization
        void Start () {
                AudioSourceMujer= Mujer.GetComponent<AudioSource>();
        //      AudioSourceMujer.clip = AudioSaludo;
        }

        // Update is called once per frame
        void OnTriggerEnter (Collider other) {

                contador = contador + 1;

                if (contador == 1) {
                        AudioSourceMujer.Play ();
                        Debug.Log ("hola man");
                }
        }

}
```

```
namespace VRTK
{
    using System.Collections;
    using System.Collections.Generic;
    using UnityEngine;
    using UnityEngine.SceneManagement;

    public class IrMenuRepeticion : MonoBehaviour
    {



        // Use this for initialization
        void Start()
        {

        }
        void OnTriggerEnter(Collider collider)
        {
            SceneManager.LoadScene("Final Tutorial");
        }
    }
}
```

```
namespace VRTK
{
    using System.Collections;
    using System.Collections.Generic;
    using UnityEngine;
    using UnityEngine.SceneManagement;

    public class IrMenuRepeticion1 : MonoBehaviour
    {



        // Use this for initialization
        void Start()
        {

        }
        void OnTriggerEnter(Collider collider)
        {
            SceneManager.LoadScene("Final Main scene");
        }
    }
}
```

```
namespace VRTK
{
    using System.Collections;
    using System.Collections.Generic;
    using UnityEngine;
    using UnityEngine.SceneManagement;

    public class IrMenuRepeticion2 : MonoBehaviour
    {



        // Use this for initialization
        void Start()
        {

        }
        void OnTriggerEnter(Collider collider)
        {
            SceneManager.LoadScene("Final Main sceneL2");
        }
    }
}
```

```
namespace VRTK
{
    using System.Collections;
    using System.Collections.Generic;
    using UnityEngine;
    using UnityEngine.SceneManagement;

    public class IrMenuRepeticion3 : MonoBehaviour
    {



        // Use this for initialization
        void Start()
        {

        }
        void OnTriggerEnter(Collider collider)
        {
            SceneManager.LoadScene("Final Main sceneL3");
        }
    }
}
```

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


public class LevelManager : MonoBehaviour {

        // Use this for initialization
        void Start () {

        }

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

        }
    public void CargaNivel(string NombreNivel)
    {
        SceneManager.LoadScene(NombreNivel);
    }
}
```

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

public class ListaCinco : MonoBehaviour {

        public Text Lista;
        private string[] listas = {
                "Ajo\nVinagre\nSal\nNaranja\nGambas",
                "Arroz\nPollo\nGuisantes\nAceite\nCebolla",
                "Lasaña\nBistec\nMantequilla\nAlmendras\nGarbanzos",
         "Manzana\nPizza\nLeche\nAzúcar\nMacarrones",
         "Sal\nPan\nPera\nTomate\nHarina",
         "Pan bimbo\nMermelada\nHelado\nLentejas\nPepino"
    };

        public int pet;

        // Use this for initialization
        void Start () {

                pet =Random.Range(0,listas.Length);
                print (listas[pet]);

                Lista.text = listas [pet];

        }

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

        }
}
```

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

public class ListaSiete : MonoBehaviour {

        public Text Lista;
        private string[] listas = {

        "Pan\nMantequilla\nLentejas\nTomate\nZanahoria\nSal\nAceite",
                "Helado\nBistec\nGarbanzos\nAzúcar\nLeche\nPatata\nPasta
de dientes",
                "Pan
bimbo\nTampax\nMermelada\nMelocotón\nPollo\nPizza\nPimienta",
         "Gambas\nVinagre\nHarina\nArroz\nCerezas\nMiel\nPiña",
         "Limón\nAlubias\nSandía\nCafé\nDesodorante\nFresas\nAlmendras"
    };
        public int pet;

        // Use this for initialization
        void Start () {

                pet =Random.Range(0,listas.Length);
                print (listas[pet]);

                Lista.text = listas [pet];

        }

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

        }
}
```

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

public class ListaTres : MonoBehaviour {

        public Text Lista;
        private string[] listas = {
            "Garbanzos\nAceite\nSal",
                "Leche\nPan\nPlátano",
                "Pollo\nGambas\nManzana",
         "Bistec\nPizza\nSandía",
         "Lentejas\nMiel\nTomate",
         "Pepino\nVinagre\nMelocotón",
         "Azúcar\nMantequilla\nHelado",
         "Arroz\nPan bimbo\nGuisantes"
    };
        public int pet;

        // Use this for initialization
        void Start () {

                pet =Random.Range(0,listas.Length);
                print (listas[pet]);

                Lista.text = listas [pet];


        }

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

        }
}
```

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

public class LoadSceneOnClick : MonoBehaviour {

    public void LoadByIndex(int sceneIndex)
    {
        SceneManager.LoadScene(sceneIndex);
    }

}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using MySql.Data.MySqlClient;
using UnityEngine.SceneManagement;



public class Login : MonoBehaviour
{
    public InputField usuarioTxt;
    public InputField contraseñaTxt;

    public void Logear()
    {
        string _log = "`usuarios` WHERE `usuario` LIKE '" +
usuarioTxt.text + "'AND `pass` LIKE'" + contraseñaTxt.text + "'";

        AdminMYSQL _adminMYSQL =
GameObject.Find("AdministradorBaseDeDatos").GetComponent<AdminMYSQL>();
        MySqlDataReader Resultado = _adminMYSQL.Select(_log);

        if (Resultado.HasRows)
        {
            Debug.Log("Login Correcto");
            Resultado.Close();
            SceneManager.LoadScene("MENU 1");
        }
        else
        {
            Debug.Log("Usuario o contraseña incorrectos");
            Resultado.Close();

        }
    }


}
```

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

public class OpenInitialDoors : MonoBehaviour {
    public GameObject flag;
    public GameObject door1;
        public GameObject door2;
        Animator animation1;
        Animator animation2;
        // Use this for initialization
        void Start () {
                animation1 = door1.GetComponent<Animator>();
                animation2 = door2.GetComponent<Animator>();
        }


        void OnTriggerEnter (Collider collider)
        {
         if (flag.GetComponent<Renderer>().material.color == Color.green)
         {
             animation1.SetBool("open", true);
             animation2.SetBool("open", true);
         }

        }
}
```

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

public class PreviousScene : MonoBehaviour {

    private static string lastLevel;

    public static void setLastLevel(string level)
    {
        lastLevel = level;
    }

    public static string getLastLevel()
    {
        return lastLevel;
    }

    public static void changeToPreviousLvl()
    {
        Application.LoadLevel(lastLevel);
    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;
using MySql.Data.MySqlClient;


public class ProductosCestaController : MonoBehaviour
{

    public int nObjects = 3;
    public bool[] objectState;
    public GameObject ObjectToDisable;
    public GameObject Contador;
    private bool all;

    private int naciertos = 0;

    public GameObject contador;
    public int timeint;
    public GameObject time;

    private void Start()
    {

        Contador _contador = contador.GetComponent<Contador>();

        Contador.SetActive(false);
        ObjectToDisable.SetActive(false);

        objectState = new bool[nObjects];
        for (int i = 0; i < nObjects; i++) {
            objectState[i] = false;
        }

    }

    public void SetObjectTrue(int id) {
        objectState[id] = true;
        Debug.Log("Set Active" + id);
        naciertos++;
        Contador _contador = contador.GetComponent<Contador>();
        _contador.Aciertos(naciertos);
        CheckAllState();




    }

    public void CheckAllState()
    {
        all = true;
        for (int i = 0; i < nObjects; i++)
        {
```

```
            if (!objectState[i])
            {
                all = false;
            }
        }

        if (all)
        {
            ObjectToDisable.SetActive(true);
            Contador.SetActive(true);
            time.SetActive(false);

        }
        else
        {
            ObjectToDisable.SetActive(false);
            Contador.SetActive(false);
        }
    }

    void Update()
    {
        timeint = (int)time.GetComponent<TimeCounter>().time;
    }


}
```

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

public class quitarcesta : MonoBehaviour {
    public GameObject cesta;
    private Collider cestita;
        // Use this for initialization
        void Start ()
    {
         cestita = cesta.GetComponent<Collider>();
         cestita.enabled = false;
        }
}
```

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

public class QuitScene : MonoBehaviour {

    public void Quit()
    {
#if UNITY_EDITOR
        UnityEditor.EditorApplication.isPlaying = false;
#else
        Application.Quit ();
#endif
    }
}
```

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

public class RaycastTuto : MonoBehaviour
{
    //public GameObject door1;
    //public GameObject door2;
    public GameObject totem1;
    public GameObject totem2;
    public GameObject totem3;
    //public GameObject LeftController;
    public GameObject RightController;
    private float distanceToSee = 0.0f;
    public GameObject PlanoVideoIzquierda;
    public GameObject PlanovideoDerecha;
    public GameObject PlanoTrackPad;

    AudioSource audiosourcestart;
    AudioSource audiosourceinicio;
    AudioSource audiosourceizquierda;
    AudioSource audiosourcederecha;
    AudioSource audiosourcegiro;
    public AudioClip audiostart;
    public AudioClip audioinicio;
    public AudioClip audioizquierda;
    public AudioClip audioderecha;
    public AudioClip audiogiro;
    public GameObject buttongrip;


    private int counter;
    private int counter2;
    private int counter3;
    private bool tot1 = false;
    private bool tot2 = false;
    private bool tot3 = false;
    //Animator  animation1;
    //Animator animation2;

    // Use this for initialization
    void Start()
    {
        //animation1 = door1.GetComponent<Animator>();
        //animation2 = door2.GetComponent<Animator>();
       // LeftController.SetActive(false);
        RightController.SetActive(false);
        audiosourcestart = GetComponent<AudioSource>();
        audiosourcestart.clip = audiostart;
        audiosourcestart.Play();
        StartCoroutine(Repeticion());
        StartCoroutine(iniciar());
        audiosourceizquierda = totem1.GetComponent<AudioSource>();
        audiosourceizquierda.clip = audioizquierda;
        audiosourcederecha = totem2.GetComponent<AudioSource>();
        audiosourcederecha.clip = audioderecha;
        audiosourcegiro = totem3.GetComponent<AudioSource>();
```

```
    audiosourcegiro.clip = audiogiro;

}

IEnumerator Repeticion()
{
    yield return new WaitForSeconds(15.0f);
    buttongrip.SetActive(true);
}

IEnumerator iniciar()
{
    yield return new WaitForSeconds(audiostart.length);
    StartCoroutine(imagen());
    audiosourceinicio = GetComponent<AudioSource>();
    audiosourceinicio.clip = audioinicio;
    audiosourceinicio.Play();
    StartCoroutine(tiempo());
}

private IEnumerator imagen()
{
    yield return new WaitForSeconds(3.0f);
    buttongrip.SetActive(false);
}

IEnumerator tiempo()
{
    yield return new WaitForSeconds(audioinicio.length);
    distanceToSee = 200.0f;
}

IEnumerator PlanoIzquierda()
{
    yield return new WaitForSeconds(5.0f);
    PlanoVideoIzquierda.SetActive(true);
}

IEnumerator PlanoTrackPadInicio()
{
    yield return new WaitForSeconds(11.0f);
    PlanoTrackPad.SetActive(true);
}

IEnumerator tiempototem1()
{
    yield return new WaitForSeconds(audioizquierda.length);
    totem2.SetActive(true);
    PlanoVideoIzquierda.SetActive(false);

}

IEnumerator tiempototem2()
{
    yield return new WaitForSeconds(audioderecha.length);
    totem3.SetActive(true);
    PlanovideoDerecha.SetActive(false);
```

```
    }

    IEnumerator tiempototem3()
    {
        yield return new WaitForSeconds(audiogiro.length);
        //LeftController.SetActive(true);
        RightController.SetActive(true);
        PlanoTrackPad.SetActive(false);
    }
    // Update is called once per frame
    void Update()
    {
        RaycastHit hit;


        if (Physics.Raycast(this.transform.position,
this.transform.forward, out hit, distanceToSee))
        {
            if (hit.collider.gameObject == totem1)
            {
                totem1.GetComponent<Renderer>().material.color =
Color.green;
                tot1 = true;
                counter = counter + 1;
                if (counter == 1)
                {
                    StartCoroutine(tiempototem1());
                    audiosourceizquierda.Play();
                    StartCoroutine(PlanoIzquierda());
                }
            }
            if (hit.collider.gameObject == totem2)
            {

                totem2.GetComponent<Renderer>().material.color =
Color.green;
                tot2 = true;
                counter2 = counter2 + 1;
                if (counter2 == 1)
                {
                    StartCoroutine(tiempototem2());
                    audiosourcederecha.Play();
                    PlanovideoDerecha.SetActive(true);
                }
            }
            if (hit.collider.gameObject == totem3)
            {

                totem3.GetComponent<Renderer>().material.color =
Color.green;
                tot3 = true;
                counter3 = counter3 + 1;
                if (counter3 == 1)
                {
                    StartCoroutine(tiempototem3());
                    StartCoroutine(PlanoTrackPadInicio());
                    audiosourcegiro.Play();
                }
```

```
            }


            //if (tot1 == true && tot2 == true && tot3 == true) {
            //audiosourcegiro.enabled = false;
            //totem1.SetActive(false);
            //totem2.SetActive(false);
            //totem3.SetActive(false);

            //animation1.enabled= true;
            //animation2.enabled= true;
            //LeftController.SetActive (true);
            //RightController.SetActive (true);
            //}

        }
    }
}
```

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

public class Resumen : MonoBehaviour {

    public GameObject DontDestroy;
    public GameObject DontDestroy2;
    public InputField usuarioTxt;


    private void Awake()
    {
        DontDestroyOnLoad(DontDestroy);
        DontDestroyOnLoad(DontDestroy2);
    }


}
```

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

public class salirnivel : MonoBehaviour {
    AudioSource audiosourceexito;
    public AudioClip Audioexito;
    AudioSource audiosourcemusic;
    public AudioClip Audiomusic;
    // Use this for initialization
    void Start () {
        audiosourcemusic = GetComponent<AudioSource>();
        audiosourcemusic.clip = Audiomusic;
        audiosourcemusic.Play();
        StartCoroutine(exito());
    }

    private IEnumerator exito()
    {
        yield return new WaitForSeconds(Audiomusic.length);
        audiosourceexito = GetComponent<AudioSource>();
        audiosourceexito.clip = Audioexito;
        audiosourceexito.Play();
        StartCoroutine(salida());
    }

    private IEnumerator salida()
    {
        yield return new WaitForSeconds(Audioexito.length);
        SceneManager.LoadScene("Final Comunidades 1");
    }

}
```

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

public class salirnivel2 : MonoBehaviour {
    AudioSource audiosourceexito;
    public AudioClip Audioexito;
    AudioSource audiosourcemusic;
    public AudioClip Audiomusic;
    // Use this for initialization
    void Start()
    {
        audiosourcemusic = GetComponent<AudioSource>();
        audiosourcemusic.clip = Audiomusic;
        audiosourcemusic.Play();
        StartCoroutine(exito());
    }

    private IEnumerator exito()
    {
        yield return new WaitForSeconds(Audiomusic.length);
        audiosourceexito = GetComponent<AudioSource>();
        audiosourceexito.clip = Audioexito;
        audiosourceexito.Play();
        StartCoroutine(salida());
    }

    private IEnumerator salida()
    {
        yield return new WaitForSeconds(Audioexito.length);
        SceneManager.LoadScene("Final Comunidades 2");
    }

}
```

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

public class salirnivel3 : MonoBehaviour {
    AudioSource audiosourceexito;
    public AudioClip Audioexito;
    AudioSource audiosourcemusic;
    public AudioClip Audiomusic;
    // Use this for initialization
    void Start()
    {
        audiosourcemusic = GetComponent<AudioSource>();
        audiosourcemusic.clip = Audiomusic;
        audiosourcemusic.Play();
        StartCoroutine(exito());
    }

    private IEnumerator exito()
    {
        yield return new WaitForSeconds(Audiomusic.length);
        audiosourceexito = GetComponent<AudioSource>();
        audiosourceexito.clip = Audioexito;
        audiosourceexito.Play();
        StartCoroutine(salida());
    }

    private IEnumerator salida()
    {
        yield return new WaitForSeconds(Audioexito.length);
        SceneManager.LoadScene("Final Comunidades 3");
    }
}
```

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

public class salirnivel4 : MonoBehaviour {
    AudioSource audiosourceexito;
    public AudioClip Audioexito;
    AudioSource audiosourcemusic;
    public AudioClip Audiomusic;
    // Use this for initialization
    void Start()
    {
        audiosourcemusic = GetComponent<AudioSource>();
        audiosourcemusic.clip = Audiomusic;
        audiosourcemusic.Play();
        StartCoroutine(exito());
    }

    private IEnumerator exito()
    {
        yield return new WaitForSeconds(Audiomusic.length);
        audiosourceexito = GetComponent<AudioSource>();
        audiosourceexito.clip = Audioexito;
        audiosourceexito.Play();
        StartCoroutine(salida());
    }

    private IEnumerator salida()
    {
        yield return new WaitForSeconds(Audioexito.length);
        SceneManager.LoadScene("Final Reponedor 1");
    }
}
```

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

public class salirnivel5 : MonoBehaviour {
    AudioSource audiosourceexito;
    public AudioClip Audioexito;
    AudioSource audiosourcemusic;
    public AudioClip Audiomusic;
    // Use this for initialization
    void Start()
    {
        audiosourcemusic = GetComponent<AudioSource>();
        audiosourcemusic.clip = Audiomusic;
        audiosourcemusic.Play();
        StartCoroutine(exito());
    }

    private IEnumerator exito()
    {
        yield return new WaitForSeconds(Audiomusic.length);
        audiosourceexito = GetComponent<AudioSource>();
        audiosourceexito.clip = Audioexito;
        audiosourceexito.Play();
        StartCoroutine(salida());
    }

    private IEnumerator salida()
    {
        yield return new WaitForSeconds(Audioexito.length);
        SceneManager.LoadScene("Final Reponedor 2");
    }
}
```

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

public class salirnivel6 : MonoBehaviour {
    AudioSource audiosourceexito;
    public AudioClip Audioexito;
    AudioSource audiosourcemusic;
    public AudioClip Audiomusic;
    // Use this for initialization
    void Start()
    {
        audiosourcemusic = GetComponent<AudioSource>();
        audiosourcemusic.clip = Audiomusic;
        audiosourcemusic.Play();
        StartCoroutine(exito());
    }

    private IEnumerator exito()
    {
        yield return new WaitForSeconds(Audiomusic.length);
        audiosourceexito = GetComponent<AudioSource>();
        audiosourceexito.clip = Audioexito;
        audiosourceexito.Play();
        StartCoroutine(salida());
    }

    private IEnumerator salida()
    {
        yield return new WaitForSeconds(Audioexito.length);
        SceneManager.LoadScene("Final Reponedor 3");
    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using Mono.Data.Sqlite;
using System.Data;
using System;
using UnityEngine.UI;

public class SobreEscribirDatos : MonoBehaviour {

        private GameObject GameControlDatos;
        private int ID;
        private int OV3 ;
        private int ONV3 ;
        private int T3 ;

        // Use this for initialization
        void Start () {

                GameControlDatos = GameObject.Find ("GameControl");
                ID= GameControlDatos.GetComponent<GameControl>().ID;
                OV3 = GameControlDatos.GetComponent<GameControl> ().OV3;
                ONV3 = GameControlDatos.GetComponent<GameControl>
().ONV3;
                T3 = GameControlDatos.GetComponent<GameControl> ().T3;
        }

        public void Escena1(){

                string conn = "URI=file:" + Application.dataPath +
"/plugins/BaseDatos.db"; //Path to database.
                IDbConnection dbconn;
                dbconn = (IDbConnection) new SqliteConnection(conn);
                dbconn.Open(); //Open connection to the database.

                IDbCommand dbcmd = dbconn.CreateCommand();
                string sqlQuery = "UPDATE Datos SET
ObjetosVal3='"+OV3+"',ObjetosNoVal3='"+ONV3+"',T3='"+T3+"' WHERE ID="+ID;
                dbcmd.CommandText = sqlQuery;
                IDataReader reader = dbcmd.ExecuteReader();


                reader.Close();
                reader = null;

                dbcmd.Dispose();
                dbcmd = null;

                dbconn.Close();
                dbconn = null;


        }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using Mono.Data.Sqlite;
using System.Data;
using System;
using UnityEngine.UI;

public class SobreEscribirDatos5 : MonoBehaviour {

        private GameObject GameControlDatos;
        private int ID;
        private int OV5 ;
        private int ONV5 ;
        private int T5 ;
        private int C5;

        // Use this for initialization
        void Start () {

                GameControlDatos = GameObject.Find ("GameControl");
                ID= GameControlDatos.GetComponent<GameControl>().ID;
                OV5 = GameControlDatos.GetComponent<GameControl> ().OV5;
                ONV5 = GameControlDatos.GetComponent<GameControl>
().ONV5;
                T5 = GameControlDatos.GetComponent<GameControl> ().T5;
                C5 = GameControlDatos.GetComponent<GameControl> ().C5;
        }

        public void Escena2(){

                string conn = "URI=file:" + Application.dataPath +
"/plugins/BaseDatos.db"; //Path to database.
                IDbConnection dbconn;
                dbconn = (IDbConnection) new SqliteConnection(conn);
                dbconn.Open(); //Open connection to the database.

                IDbCommand dbcmd = dbconn.CreateCommand();
                string sqlQuery = "UPDATE Datos SET
ObjetosVal5='"+OV5+"',ObjetosNoVal5='"+ONV5+"',T5='"+T5+"',Cons5='"+C5+"'
WHERE ID="+ID;
                dbcmd.CommandText = sqlQuery;
                IDataReader reader = dbcmd.ExecuteReader();


                reader.Close();
                reader = null;

                dbcmd.Dispose();
                dbcmd = null;

                dbconn.Close();
                dbconn = null;


        }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using Mono.Data.Sqlite;
using System.Data;
using System;
using UnityEngine.UI;

public class SobreEscribirDatos7 : MonoBehaviour {

        private GameObject GameControlDatos;
        private int ID;
        private int OV7 ;
        private int ONV7 ;
        private int T7 ;
    private int C7;
        // Use this for initialization
        void Start () {

                GameControlDatos = GameObject.Find ("GameControl");
                ID= GameControlDatos.GetComponent<GameControl>().ID;
                OV7 = GameControlDatos.GetComponent<GameControl> ().OV7;
                ONV7 = GameControlDatos.GetComponent<GameControl>
().ONV7;
                T7 = GameControlDatos.GetComponent<GameControl> ().T7;
         C7 = GameControlDatos.GetComponent<GameControl>().C7;
    }

        public void Escena3(){

                string conn = "URI=file:" + Application.dataPath +
"/plugins/BaseDatos.db"; //Path to database.
                IDbConnection dbconn;
                dbconn = (IDbConnection) new SqliteConnection(conn);
                dbconn.Open(); //Open connection to the database.

                IDbCommand dbcmd = dbconn.CreateCommand();
                string sqlQuery = "UPDATE Datos SET
ObjetosVal7='"+OV7+"',ObjetosNoVal7='"+ONV7+"',T7='"+T7+"',Cons7='"+C7+"'
WHERE ID="+ID;
                dbcmd.CommandText = sqlQuery;
                IDataReader reader = dbcmd.ExecuteReader();


                reader.Close();
                reader = null;

                dbcmd.Dispose();
                dbcmd = null;

                dbconn.Close();
                dbconn = null;


        }
}
```

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

public class soundsrepeat : MonoBehaviour
{
    private SteamVR_TrackedController controller;
    private AudioSource audiosource;


        // Use this for initialization
        //void Start ()
    //{
        //audiosource = GetComponent<AudioSource>();

        //controller = GetComponentInParent<SteamVR_TrackedController>();
        //controller.MenuButtonClicked = audiosource;
    //}


    //private void audiosource(object sender);

    //{
        //audiosource.Play();
    //}

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

        }
}
```

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

public class StartTalking : MonoBehaviour
{

    public GameObject Mujer;
    public GameObject LeftController;
    public GameObject RightController;
    AudioSource AudioSourceMujer;
    public AudioClip audiodistrac;
    int contador = 0;
    // Use this for initialization
    void Start()
    {
        AudioSourceMujer = Mujer.GetComponent<AudioSource>();
    }

    void OnTriggerEnter(Collider collider)
    {

        LeftController.SetActive(false);
        RightController.SetActive(false);
        StartCoroutine(TiempoEsperar());
        contador = contador + 1;

        if (contador == 1)
        {
            AudioSourceMujer.Play();
        }
    }
    IEnumerator TiempoEsperar()
    {
        yield return new WaitForSeconds(audiodistrac.length);
        LeftController.SetActive(true);
        RightController.SetActive(true);
        Destroy(gameObject);
    }
}
```

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

public class StartWalking : MonoBehaviour {

        public GameObject Mujer;
        //public GameObject LeftController;
        public GameObject RightController;
        public int TiempoEspera;//Normalmente el tiempo será de 10 seg
pero por si se varía la animación
        Animator animation1;
        AudioSource AudioSourceMujer;
        public int TiempoSaludar;
    public int tiempomoverse;
    private int contador;
        private int contadorsaludo;
    public GameObject Pared;
        // Use this for initialization
        void Start () {

                animation1 = Mujer.GetComponent<Animator> ();
                AudioSourceMujer= Mujer.GetComponent<AudioSource>();
        }

        // Update is called once per frame
        void OnTriggerEnter (Collider collider) {
        contador = contador + 1;
        if (contador == 1)

        {
            animation1.SetBool("walking", false);
            //animation1.enabled = true;
            //LeftController.SetActive (false);
            RightController.SetActive(false);
            StartCoroutine(TiempoEsperar());
            StartCoroutine(TiempoSaludo());
            StartCoroutine(Tiempomovers());
            Destroy(Pared);
        }
        }
    IEnumerator Tiempomovers()
    {
        yield return new WaitForSeconds(tiempomoverse);

        RightController.SetActive(true);

    }
    IEnumerator TiempoSaludo()
    {
        yield return new WaitForSeconds(TiempoSaludar);
        contadorsaludo = contadorsaludo + 1;
        if (contadorsaludo == 1)
        {
            AudioSourceMujer.Play();
        }
    }
    IEnumerator TiempoEsperar()
```

```
        {
                yield return new WaitForSeconds(TiempoEspera);
        //      LeftController.SetActive (true);
                Mujer.SetActive (false);
         Destroy(gameObject);
        }


}
```

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

public class StopGravity : MonoBehaviour {

        public Text ObjetosCesta;
        private string ObjetosPrevios;
        private string ObjetosPrevios2;
        public GameObject referencia;
        private int V1=0;
        private int V2=0;
        private int V3=0;
        public int Valid=0;
        public int NoVal=0;
        private int pet2;
        public GameObject time;
        public int timeint;
    public GameObject señal;
    private Collider capa;
    public GameObject quitar2;
    private Collider cesta2;
    AudioSource audiosourceexito;
    public AudioClip Audioexito;
    AudioSource audiosourcemusic;
    public AudioClip Audiomusic;
    private int x = 0;

    // Use this for initialization
    void Start () {

        pet2 = referencia.GetComponent<ListaTres>().pet;
    }

    // Update is called once per frame
    void OnTriggerEnter (Collider other) {

        if (other.tag != "Untagged") {

                        other.GetComponent<Rigidbody> ();
                        //other.attachedRigidbody.isKinematic = true;
                        other.gameObject.transform.parent = transform;

                        if (pet2 == 0) {

                                if (other.tag == "Garbanzos") {
                                        V1 = 1;
                                        print (V1 + ("V1"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag == "Aceite") {
                                        V2 = 1;
                                        print (V2 + ("V2"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag == "Sal") {
```

```
                                        V3 = 1;
                                        print (V3 + ("V3"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag !="Garbanzos" && other.tag
!="Aceite" && other.tag !="Sal")
                                {
                                        NoVal = NoVal + 1;
                                        print (NoVal+("NoVal"));
                                }

                        }
                        if (pet2 == 1) {

                                if (other.tag == "Leche") {
                                        V1 = 1;
                                        print (V1 + ("V1"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag == "Pan") {
                                        V2 = 1;
                                        print (V2 + ("V2"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag == "Plátanos") {
                                        V3 = 1;
                                        print (V3 + ("V3"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag !="Leche" && other.tag
!="Pan" && other.tag !="Plátanos")
                                {
                                        NoVal = NoVal + 1;
                                        print (NoVal+("NoVal"));
                                }

                        }
                        if (pet2 == 2) {

                                if (other.tag == "Pollo") {
                                        V1 = 1;
                                        print (V1 + ("V1"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag == "Gambas") {
                                        V2 = 1;
                                        print (V2 + ("V2"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag == "Manzana") {
                                        V3 = 1;
                                        print (V3 + ("V3"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag !="Pollo" && other.tag !=
"Gambas" && other.tag !="Manzana")
                                {
```

```
                                        NoVal = NoVal + 1;
                                        print (NoVal+("NoVal"));
                                }

                        }

                        if (pet2 == 3) {

                                if (other.tag == "Bistec") {
                                        V1 = 1;
                                        print (V1 + ("V1"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag == "Pizza") {
                                        V2 = 1;
                                        print (V2 + ("V2"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag == "Sandía") {
                                        V3 = 1;
                                        print (V3 + ("V3"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag !="Bistec" && other.tag
!="Pizza" && other.tag != "Sandía")
                                {
                                        NoVal = NoVal + 1;
                                        print (NoVal+("NoVal"));
                                }

                        }
                        if (pet2 == 4) {

                                if (other.tag == "Lentejas") {
                                        V1 = 1;
                                        print (V1 + ("V1"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag == "Miel") {
                                        V2 = 1;
                                        print (V2 + ("V2"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag == "Tomate") {
                                        V3 = 1;
                                        print (V3 + ("V3"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag != "Lentejas" && other.tag
!= "Miel" && other.tag !="Tomate")
                                {
                                        NoVal = NoVal + 1;
                                        print (NoVal+("NoVal"));
                                }

                        }
                        if (pet2 == 5) {
```

```
                                if (other.tag == "Pepino") {
                                        V1 = 1;
                                        print (V1 + ("V1"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag == "Vinagre") {
                                        V2 = 1;
                                        print (V2 + ("V2"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag == "Melocotón") {
                                        V3 = 1;
                                        print (V3 + ("V3"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag != "Pepino" && other.tag
!="Vinagre" && other.tag != "Melocotón")
                                {
                                        NoVal = NoVal + 1;
                                        print (NoVal+("NoVal"));
                                }

                        }
                        if (pet2 == 6) {

                                if (other.tag == "Azúcar") {
                                        V1 = 1;
                                        print (V1 + ("V1"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag == "Mantequilla") {
                                        V2 = 1;
                                        print (V2 + ("V2"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag == "Helado") {
                                        V3 = 1;
                                        print (V3 + ("V3"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag != "Azúcar" && other.tag !=
"Mantequilla" && other.tag != "Helado")
                                {
                                        NoVal = NoVal + 1;
                                        print (NoVal+("NoVal"));
                                }

                        }
                        if (pet2 == 7) {

                                if (other.tag == "Arroz") {
                                        V1 = 1;
                                        print (V1 + ("V1"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag == "Pan bimbo") {
                                        V2 = 1;
```

```
                                        print (V2 + ("V2"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag == "Guisantes") {
                                        V3 = 1;
                                        print (V3 + ("V3"));
                                        print (Valid+("Valid"));
                                }
                                if (other.tag !="Arroz" && other.tag !=
"Pan bimbo" && other.tag !="Guisantes")
                                {
                                        NoVal = NoVal + 1;
                                        print (NoVal+("NoVal"));
                                }
                        }

                        other.gameObject.SetActive (false);

            ObjetosCesta.text=
string.Format("\n"+other.tag+ObjetosPrevios2);
                        ObjetosPrevios= ObjetosCesta.text;
                        ObjetosPrevios2= ObjetosPrevios;
        }
        }

        void Update (){

        capa = señal.GetComponent<Collider>();

        if (capa.enabled == true)
        {
            cesta2 = quitar2.GetComponent<Collider>();
            cesta2.enabled = true;
        }
        if (capa.enabled == false)
        {
            cesta2 = quitar2.GetComponent<Collider>();
            cesta2.enabled = false;
        }
        if ((Valid == 3) & (x == 0))
        {
            x = 1;
            audiosourcemusic = GetComponent<AudioSource>();
            audiosourcemusic.clip = Audiomusic;
            audiosourcemusic.Play();
            StartCoroutine(exito());
        }
        Valid = V1 + V2 + V3;
                timeint= (int)time.GetComponent<TimeCounter>().time;

                }

    private IEnumerator exito()
    {
        yield return new WaitForSeconds(Audiomusic.length);
        audiosourceexito = GetComponent<AudioSource>();
        audiosourceexito.clip = Audioexito;
        audiosourceexito.Play();
```

```
        }
}
```

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

public class StopGravity5 : MonoBehaviour {

        public Text ObjetosCesta;
        private string ObjetosPrevios;
        private string ObjetosPrevios2;
        public GameObject referencia;
        private int V1=0;
        private int V2= 0;
        private int V3= 0;
        private int V4= 0;
        private int V5= 0;
        private int pet2;
        public int Valid=0;
        public int NoVal=0;
    public GameObject señal;
    private Collider capa;
    public GameObject quitar2;
    private Collider cesta2;
    public GameObject time;
    public int timeint;
    AudioSource audiosourceexito;
    public AudioClip Audioexito;
    AudioSource audiosourcemusic;
    public AudioClip Audiomusic;
    private int x = 0;

    // Use this for initialization
    void Start () {

                pet2 = referencia.GetComponent<ListaCinco> ().pet;

        }

        void OnTriggerEnter (Collider other) {

                if (other.tag != "Untagged") {

                        other.GetComponent<Rigidbody> ();
                        //other.attachedRigidbody.isKinematic = true;
                        other.gameObject.transform.parent = transform;

                        if (pet2 == 0) {

                                if (other.tag == "Ajo") {
                                        V1 = 1;
                                        print (V1);
                                }
                                if (other.tag == "Vinagre") {
                                        V2 = 1;
                                        print (V2);
                                }
                                if (other.tag == "Sal") {
                                        V3 = 1;
```

```
                                        print (V3);
                                }
                                if (other.tag == "Naranja") {
                                        V4 = 1;
                                        print (V4);
                                }
                                if (other.tag == "Gambas") {
                                        V5 = 1;
                                        print (V5);
                                }
                                if (other.tag != "Ajo" && other.tag
!="Vinagre" && other.tag !="Sal" && other.tag != "Naranja" && other.tag
!= "Gambas")
                                {
                                        NoVal = NoVal + 1;
                                        print (NoVal+("NoVal"));
                                }
                        }

                        if (pet2 == 1) {

                                if (other.tag == "Arroz") {
                                        V1 = 1;
                                        print (V1);
                                }
                                if (other.tag == "Pollo") {
                                        V2 = 1;
                                        print (V2);
                                }
                                if (other.tag == "Guisantes") {
                                        V3 = 1;
                                        print (V3);
                                }
                                if (other.tag == "Aceite") {
                                        V4 = 1;
                                        print (V4);
                                }
                                if (other.tag == "Cebolla") {
                                        V5 = 1;
                                        print (V5);
                                }
                                if (other.tag !="Arroz" && other.tag
!="Pollo" && other.tag !="Guisantes" && other.tag != "Aceite" &&
other.tag !="Cebolla")
                                {
                                        NoVal = NoVal + 1;
                                        print (NoVal+("NoVal"));
                                }
                        }

                        if (pet2 == 2) {

                                if (other.tag == "Lasaña") {
                                        V1 = 1;
                                        print (V1);
                                }
                                if (other.tag == "Bistec") {
```

```
                                    V2 = 1;
                                    print (V2);
                                }
                                if (other.tag == "Mantequilla") {
                                    V3 = 1;
                                    print (V3);
                                }
                                if (other.tag == "Almendras") {
                                    V4 = 1;
                                    print (V4);
                                }
                                if (other.tag == "Garbanzos") {
                                    V5 = 1;
                                    print (V5);
                                }
                                if (other.tag != "Lasaña" && other.tag
!="Bistec" && other.tag != "Mantequilla" && other.tag != "Almendras" &&
other.tag !="Garbanzos")
                                {
                                    NoVal = NoVal + 1;
                                    print (NoVal+("NoVal"));
                                }
                            }

            if (pet2 == 3)
            {

                if (other.tag == "Manzana")
                {
                    V1 = 1;
                    print(V1);
                }
                if (other.tag == "Pizza")
                {
                    V2 = 1;
                    print(V2);
                }
                if (other.tag == "Leche")
                {
                    V3 = 1;
                    print(V3);
                }
                if (other.tag == "Azúcar")
                {
                    V4 = 1;
                    print(V4);
                }
                if (other.tag == "Macarrones")
                {
                    V5 = 1;
                    print(V5);
                }
                if (other.tag != "Manzana" && other.tag != "Pizza" &&
other.tag != "Leche" && other.tag != "Azúcar" && other.tag !=
"Macarrones")
                {
                    NoVal = NoVal + 1;
                    print(NoVal + ("NoVal"));
```

```
            }
        }

        if (pet2 == 4)
        {
            if (other.tag == "Sal")
            {
                V1 = 1;
                print(V1);
            }
            if (other.tag == "Pan")
            {
                V2 = 1;
                print(V2);
            }
            if (other.tag == "Pera")
            {
                V3 = 1;
                print(V3);
            }
            if (other.tag == "Tomate")
            {
                V4 = 1;
                print(V4);
            }
            if (other.tag == "Harina")
            {
                V5 = 1;
                print(V5);
            }
            if (other.tag != "Sal" && other.tag != "Pan" && other.tag
!= "Pera" && other.tag != "Tomate" && other.tag != "Harina")
            {
                NoVal = NoVal + 1;
                print(NoVal + ("NoVal"));
            }
        }
        if (pet2 == 5)
        {

            if (other.tag == "Pan bimbo")
            {
                V1 = 1;
                print(V1);
            }
            if (other.tag == "Mermelada")
            {
                V2 = 1;
                print(V2);
            }
            if (other.tag == "Helado")
            {
                V3 = 1;
                print(V3);
            }
            if (other.tag == "Lentejas")
            {
```

```
                    V4 = 1;
                    print(V4);
                }
                if (other.tag == "Pepino")
                {
                    V5 = 1;
                    print(V5);
                }
                if (other.tag != "Pan bimbo" && other.tag != "Mermelada"
&& other.tag != "Helado" && other.tag != "Lentejas" && other.tag !=
"Pepino")
                {
                    NoVal = NoVal + 1;
                    print(NoVal + ("NoVal"));
                }
            }

            other.gameObject.SetActive (false);

                        ObjetosCesta.text=
string.Format("\n"+other.tag+ObjetosPrevios2);
                        ObjetosPrevios= ObjetosCesta.text;
                        ObjetosPrevios2= ObjetosPrevios;
                }
        }

        // Update is called once per frame
        void Update () {
        capa = señal.GetComponent<Collider>();

        if (capa.enabled == true)
        {
            cesta2 = quitar2.GetComponent<Collider>();
            cesta2.enabled = true;
        }
        if (capa.enabled == false)
        {
            cesta2 = quitar2.GetComponent<Collider>();
            cesta2.enabled = false;
        }
        if ((Valid == 5) & (x == 0))
        {
            x = 1;
            audiosourcemusic = GetComponent<AudioSource>();
            audiosourcemusic.clip = Audiomusic;
            audiosourcemusic.Play();
            StartCoroutine(exito());
        }
        Valid = V1 + V2 + V3 + V4 + V5;
                timeint= (int)time.GetComponent<TimeCounter>().time;
        }

    private IEnumerator exito()
    {
        yield return new WaitForSeconds(Audiomusic.length);
        audiosourceexito = GetComponent<AudioSource>();
        audiosourceexito.clip = Audioexito;
        audiosourceexito.Play();
```

```
    }

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

public class StopGravity7 : MonoBehaviour {

        public Text ObjetosCesta;
        private string ObjetosPrevios;
        private string ObjetosPrevios2;
        public GameObject referencia;
        private int V1=0;
        private int V2= 0;
        private int V3= 0;
        private int V4= 0;
        private int V5= 0;
        private int V6= 0;
        private int V7= 0;
        private int pet2;
        public int Valid=0;
        public int NoVal=0;
        public GameObject time;
        public int timeint;
    public GameObject señal;
    private Collider capa;
    public GameObject quitar2;
    private Collider cesta2;
    AudioSource audiosourceexito;
    public AudioClip Audioexito;
    AudioSource audiosourcemusic;
    public AudioClip Audiomusic;
    private int x = 0;

    // Use this for initialization
    void Start () {

                pet2 = referencia.GetComponent<ListaSiete> ().pet;

        }

        void OnTriggerEnter (Collider other) {

                if (other.tag != "Untagged") {

                        other.GetComponent<Rigidbody> ();
                        //other.attachedRigidbody.isKinematic = true;
                        other.gameObject.transform.parent = transform;

                        if (pet2 == 0) {

                                if (other.tag == "Pan") {
                                        V1 = 1;
                                        print (V1);
                                }
                                if (other.tag == "Mantequilla") {
```

```
                                V2 = 1;
                                print (V2);
                        }
                        if (other.tag == "Lentejas") {
                                V3 = 1;
                                print (V3);
                        }
                        if (other.tag == "Tomate") {
                                V4 = 1;
                                print (V4);
                        }
                        if (other.tag == "Zanahoria") {
                                V5 = 1;
                                print (V5);
                        }
                        if (other.tag == "Sal") {
                                V6 = 1;
                                print (V5);
                        }
                        if (other.tag == "Aceite") {
                                V7 = 1;
                                print (V5);
                        }
                        if (other.tag != "Pan" && other.tag !=
"Mantequilla" && other.tag != "Lentejas" && other.tag !="Tomate" &&
other.tag !="Zanahoria" && other.tag !="Sal" && other.tag !="Aceite")
                        {
                                NoVal = NoVal + 1;
                                print (NoVal+("NoVal"));
                        }
                }

                if (pet2 == 1) {

                        if (other.tag == "Helado") {
                                V1 = 1;
                                print (V1);
                        }
                        if (other.tag == "Bistec") {
                                V2 = 1;
                                print (V2);
                        }
                        if (other.tag == "Garbanzos") {
                                V3 = 1;
                                print (V3);
                        }
                        if (other.tag == "Azúcar") {
                                V4 = 1;
                                print (V4);
                        }
                        if (other.tag == "Leche") {
                                V5 = 1;
                                print (V5);
                        }
                        if (other.tag == "Patata") {
                                V6 = 1;
                                print (V5);
                        }
```

```
                                if (other.tag == "Pasta de dientes") {
                                        V7 = 1;
                                        print (V5);
                                }
                                if (other.tag !="Helado" && other.tag
!="Bistec" && other.tag !="Garbanzos" && other.tag != "Azúcar" &&
other.tag != "Leche" && other.tag !="Patata" && other.tag != "Pasta de
dientes")
                                {
                                        NoVal = NoVal + 1;
                                        print (NoVal+("NoVal"));
                                }
                        }
                        if (pet2 == 2) {

                                if (other.tag == "Pan bimbo") {
                                        V1 = 1;
                                        print (V1);
                                }
                                if (other.tag == "Tampax") {
                                        V2 = 1;
                                        print (V2);
                                }
                                if (other.tag == "Mermelada") {
                                        V3 = 1;
                                        print (V3);
                                }
                                if (other.tag == "Melocotón") {
                                        V4 = 1;
                                        print (V4);
                                }
                                if (other.tag == "Pollo") {
                                        V5 = 1;
                                        print (V5);
                                }
                                if (other.tag == "Pizza") {
                                        V6 = 1;
                                        print (V5);
                                }
                                if (other.tag == "Pimienta") {
                                        V7 = 1;
                                        print (V5);
                                }
                                if (other.tag !="Pan bimbo" && other.tag
!= "Tampax" && other.tag !="Mermelada" && other.tag != "Melocotón" &&
other.tag != "Pollo" && other.tag != "Pizza" && other.tag != "Pimienta")
                                {
                                        NoVal = NoVal + 1;
                                        print (NoVal+("NoVal"));
                                }
                        }
            if (pet2 == 3)
            {

                if (other.tag == "Gambas")
                {
                    V1 = 1;
```

```
                    print(V1);
                }
                if (other.tag == "Vinagre")
                {
                    V2 = 1;
                    print(V2);
                }
                if (other.tag == "Harina")
                {
                    V3 = 1;
                    print(V3);
                }
                if (other.tag == "Arroz")
                {
                    V4 = 1;
                    print(V4);
                }
                if (other.tag == "Cerezas")
                {
                    V5 = 1;
                    print(V5);
                }
                if (other.tag == "Miel")
                {
                    V6 = 1;
                    print(V5);
                }
                if (other.tag == "Piña")
                {
                    V7 = 1;
                    print(V5);
                }
                if (other.tag != "Gambas" && other.tag != "Vinagre" &&
other.tag != "Harina" && other.tag != "Arroz" && other.tag != "Cerezas"
&& other.tag != "Miel" && other.tag != "Piña")
                {
                    NoVal = NoVal + 1;
                    print(NoVal + ("NoVal"));
                }
            }
            if (pet2 == 4)
            {

                if (other.tag == "Limón")
                {
                    V1 = 1;
                    print(V1);
                }
                if (other.tag == "Alubias")
                {
                    V2 = 1;
                    print(V2);
                }
                if (other.tag == "Sandía")
                {
                    V3 = 1;
                    print(V3);
                }
```

```
                if (other.tag == "Café")
                {
                    V4 = 1;
                    print(V4);
                }
                if (other.tag == "Desodorante")
                {
                    V5 = 1;
                    print(V5);
                }
                if (other.tag == "Fresas")
                {
                    V6 = 1;
                    print(V5);
                }
                if (other.tag == "Almendras")
                {
                    V7 = 1;
                    print(V5);
                }
                if (other.tag != "Limón" && other.tag != "Alubias" &&
other.tag != "Sandía" && other.tag != "Café" && other.tag !=
"Desodorante" && other.tag != "Fresas" && other.tag != "Almendras")
                {
                    NoVal = NoVal + 1;
                    print(NoVal + ("NoVal"));
                }
            }
            other.gameObject.SetActive (false);

                        ObjetosCesta.text=
string.Format("\n"+other.tag+ObjetosPrevios2);
                        ObjetosPrevios= ObjetosCesta.text;
                        ObjetosPrevios2= ObjetosPrevios;
                }
        }

        // Update is called once per frame
        void Update () {
         capa = señal.GetComponent<Collider>();

         if (capa.enabled == true)
         {
             cesta2 = quitar2.GetComponent<Collider>();
             cesta2.enabled = true;
         }
         if (capa.enabled == false)
         {
             cesta2 = quitar2.GetComponent<Collider>();
             cesta2.enabled = false;
         }
         if ((Valid == 7) & (x == 0))
         {
             x = 1;
             audiosourcemusic = GetComponent<AudioSource>();
             audiosourcemusic.clip = Audiomusic;
             audiosourcemusic.Play();
```

```
            StartCoroutine(exito());
        }
        Valid = V1 + V2 + V3 + V4 + V5 + V6 + V7;
               timeint= (int)time.GetComponent<TimeCounter>().time;
        }

    private IEnumerator exito()
    {
        yield return new WaitForSeconds(Audiomusic.length);
        audiosourceexito = GetComponent<AudioSource>();
        audiosourceexito.clip = Audioexito;
        audiosourceexito.Play();
    }
}
```

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

public class TimeCounter : MonoBehaviour {
        public Text Timer;
        public float time= 0.0f;
        // Use this for initialization

        void Start () {

        }

        // Update is called once per frame
        void Update () {
                time += Time.deltaTime;

                var minutes = (int)time / 60;
                var seconds = time % 60;
                var fraction = (time * 100) % 100;

                Timer.text = string.Format("{0:00} : {1:00} ", minutes,
seconds);

                //Timer.text = string.Format("{0:00} : {1:00} : {2:000}",
minutes, seconds, fraction);
        }
}
```

```
using System.Collections;
using UnityEngine.UI;
using UnityEngine;
using UnityEngine.SceneManagement;
using MySql.Data.MySqlClient;

public class timer : MonoBehaviour {
    public GameObject DesactivaContador;
    public GameObject ObjectToDisable;
    public GameObject Contador;
    public Text contador;
    public float tiempo = 30f;




    void Start ()
    {
        contador.text = " " + tiempo;


    }

    // Update is called once per frame
    void Update()
    {
        tiempo -= Time.deltaTime;
        contador.text = " " + tiempo.ToString("f0");

        if (tiempo <= 0)
        {
            contador.text = "0";
            DesactivaContador.SetActive(false);
            ObjectToDisable.SetActive(true);
            Contador.SetActive(true);
        }

     }

    }
```

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

public class TimerRecogida : MonoBehaviour {



                public GameObject time;
                public int timeint;


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

                timeint = (int)time.GetComponent<TimeCounter> ().time;

        }

}
```