Use unity to help patients recover

Background:

Virtual environment modeling technology of upper limb rehabilitation training robot In order to change the situation of relying solely on hands training of therapists in stroke rehabilitation training, researchers have applied robot technology to the field of stroke rehabilitation. Research shows that if we can provide various forms of information feedback during training, give full play to the subjective initiative of patients, and give hints or suggestions to patients according to their status, the rehabilitation effect will be greatly improved. Therefore, researchers have designed a rehabilitation training system based on virtual reality to stimulate patients' interest and motivation in rehabilitation training. The games we make are mainly used for upper limbs.

Principles of Virtual Reality (VR) Technology Therapy

VR system is a multi-dimensional virtual sensory environment partially or completely generated by computer, which generates various sensory information for participants. Three dimensional stereoscopic display is an essential key equipment, which is the main means for the system to output feedback information to users.

The introduction of this project is divided into the following parts:

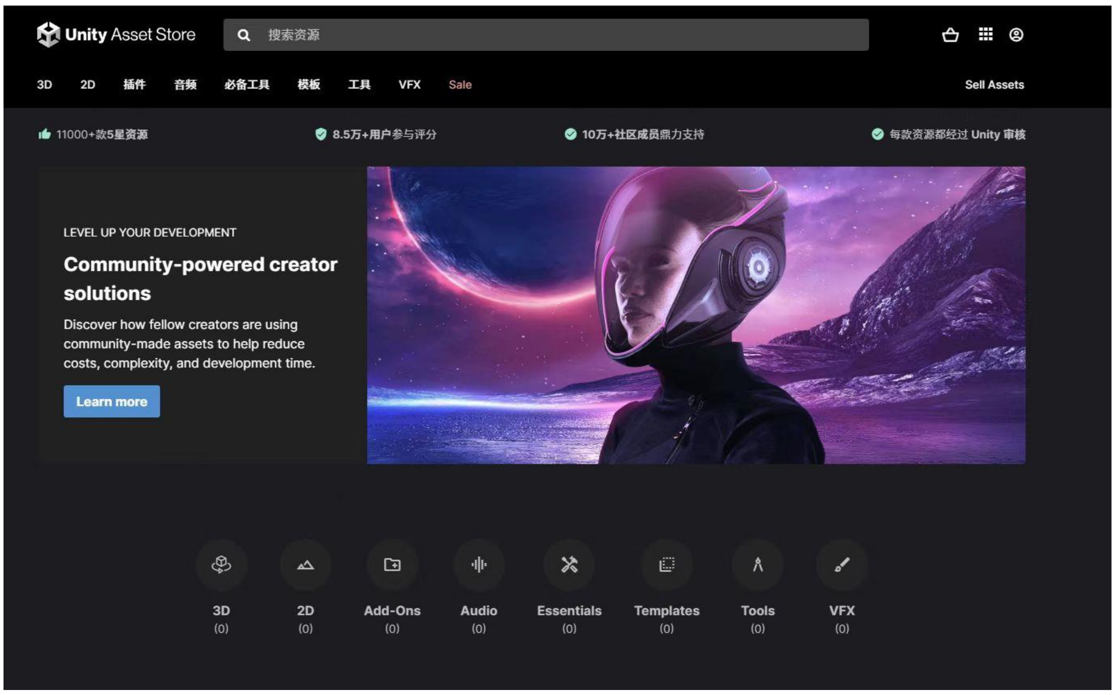
1. **Environment Construction**
2. **Game Control Code Writing**
3. **VR Connection, and Data Collection**
4. **Data Collection**
5. **Team Members**

1. Environment Construction

In order to bring a relaxed and happy atmosphere to the rehabilitation of patients, we adopted a bright style in the scene design, and also added cheerful music to increase the atmosphere. I

Time: 00:00:00

In the process of setting up the scene, because the design time is relatively short, I mainly use the model downloaded from the unit asset to open the official website. After downloading the model from the website, we can use it in unity.

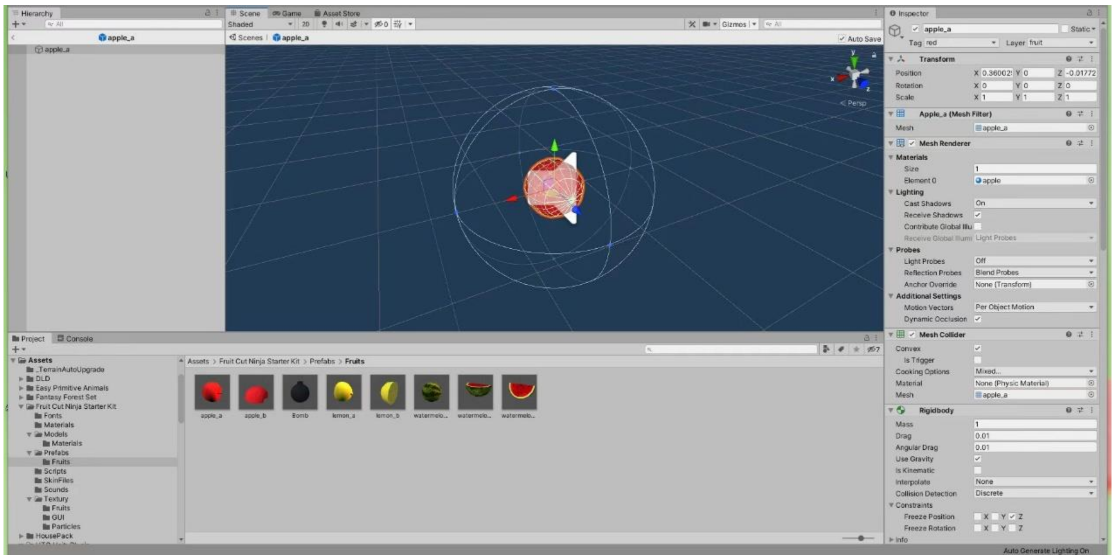


LEVEL UP YOUR DEVELOPMENT

Community-powered creator solutions

Discover how fellow creators are using community-made assets to help reduce costs, complexity, and development time.

The following is about the fruit. We use 3Dmax to model the fruit here. When modeling, we have modeled the overall appearance and separate appearance of the fruit. At the same time, we use the fireworks effect in Unity to design the fruit to be sprayed when cutting

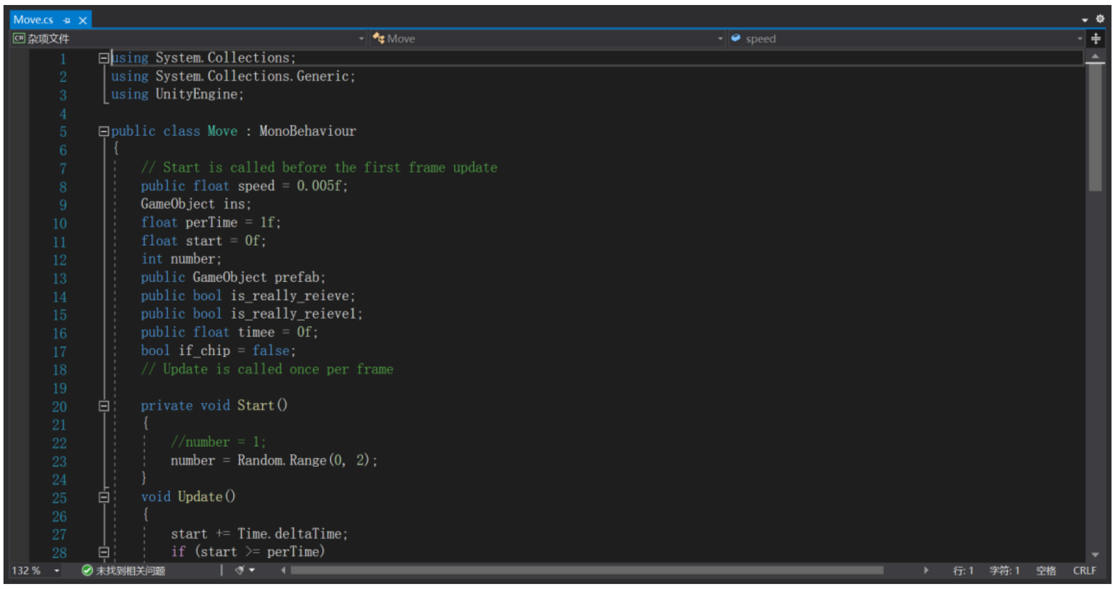


```
Inspector
[✓] apple_a                         Static
Tag red                Layer fruit
Transform
Position    X 0.36002  Y 0   Z -0.01772
Rotation    X 0        Y 0   Z 0
Scale       X 1        Y 1   Z 1
Apple_a (Mesh Filter)
Mesh                     apple_a
[✓] Mesh Renderer
Materials
  Size                   1
  Element 0              apple
Lighting
  Cast Shadows           On
  Receive Shadows        ✓
  Contribute Global Illu
  Receive Global Illumi  Light Probes
Probes
  Light Probes           Off
  Reflection Probes      Blend Probes
  Anchor Override        None (Transform)
Additional Settings
  Motion Vectors         Per Object Motion
  Dynamic Occlusion      ✓
[✓] Mesh Collider
Convex                   ✓
  Is Trigger
Cooking Options          Mixed...
Material                 None (Physic Material)
Mesh                     apple_a
Rigidbody
Mass                     1
Drag                     0.01
Angular Drag             0.01
Use Gravity              ✓
Is Kinematic
Interpolate              None
Collision Detection      Discrete
Constraints
  Freeze Position        X  Y  [✓ Z]
  Freeze Rotation        X  Y  Z
Info
Auto Generate Lighting On
```

2. Game Control Code Writing

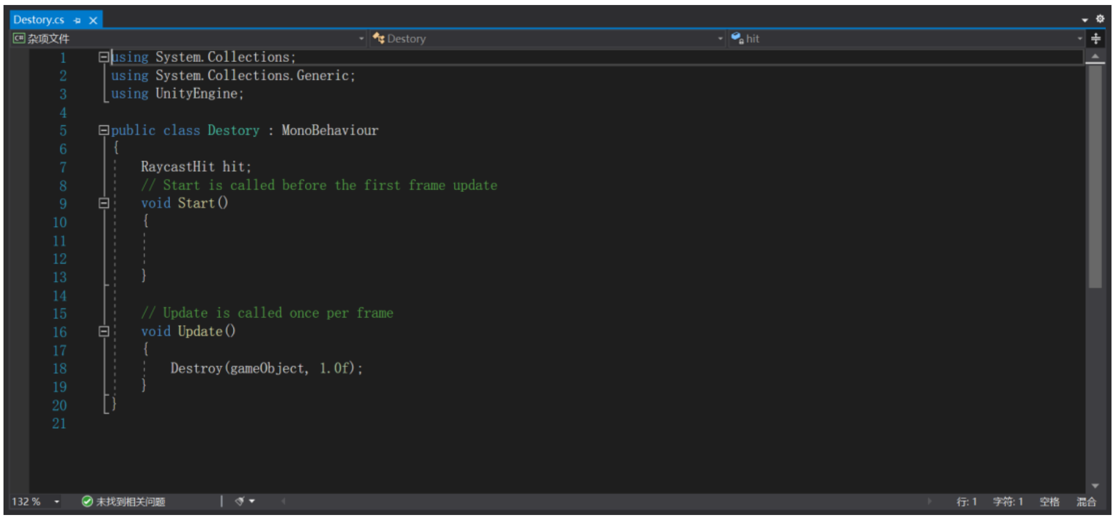
The first script is Move, which is mainly hung on fruit. The main purpose is to control the direction and speed of fruit movement at the beginning.

Firstly, we use a random function to control the left and right movement of the fruit. The time function here is used to pause the newly generated fruit for a few seconds to give players enough reaction time. Then, after receiving the left and right numbers, give four seconds for the fruit movement to use the tanslate function. After four seconds of movement, set the acceleration of the fruit to 0, and let the fruit stop at the corresponding position to ensure that the fruit can be cut.

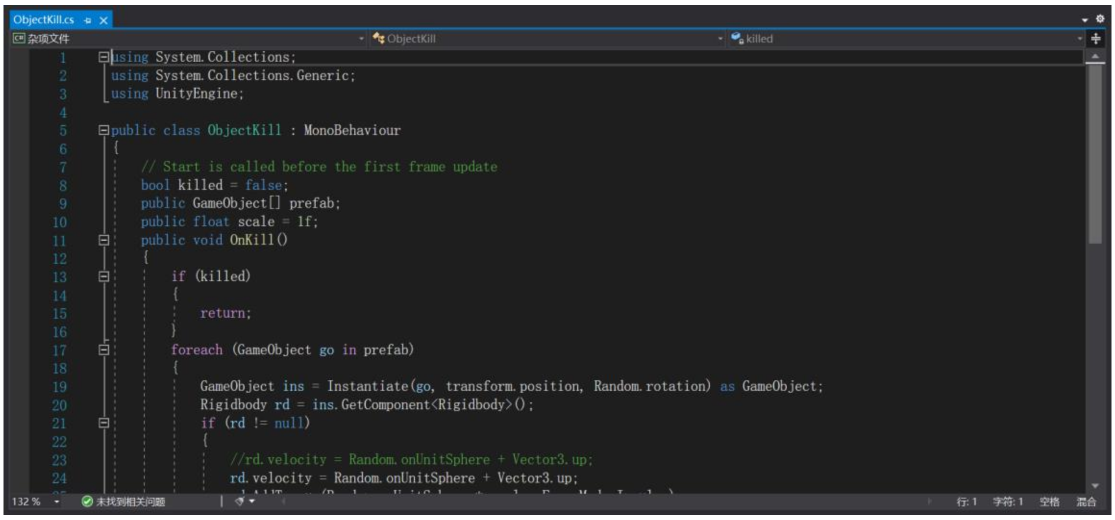


```csharp
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Move : MonoBehaviour
{
    // Start is called before the first frame update
    public float speed = 0.005f;
    GameObject ins;
    float perTime = 1f;
    float start = 0f;
    int number;
    public GameObject prefab;
    public bool is_really_reieve;
    public bool is_really_reievel;
    public float timee = 0f;
    bool if_chip = false;
    // Update is called once per frame

    private void Start()
    {
        //number = 1;
        number = Random.Range(0, 2);
    }
    void Update()
    {
        start += Time.deltaTime;
        if (start >= perTime)
```

The second and third scripts are Destory and ObjectKill, which are also used to control fruits. When we cut the fruit, we need to delete the fruit to ensure the beauty of the overall picture. At the same time, when cutting the left and right fruits, we also need to receive the instruction of the hammer to drop when the player misjudges, and then the fruit will disappear.

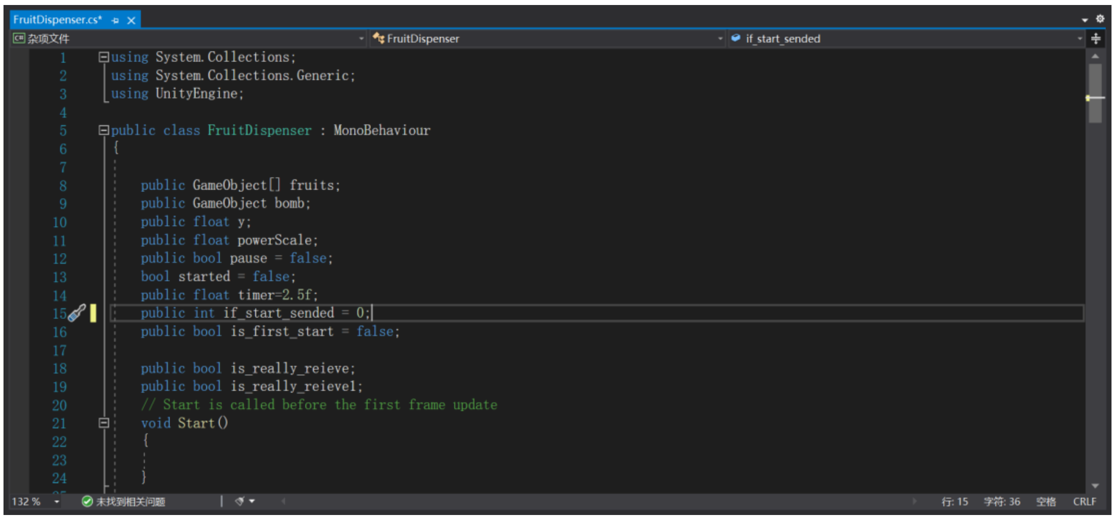


```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Destory : MonoBehaviour
{
    RaycastHit hit;
    // Start is called before the first frame update
    void Start()
    {


    }

    // Update is called once per frame
    void Update()
    {
        Destroy(gameObject, 1.0f);
    }
}
```

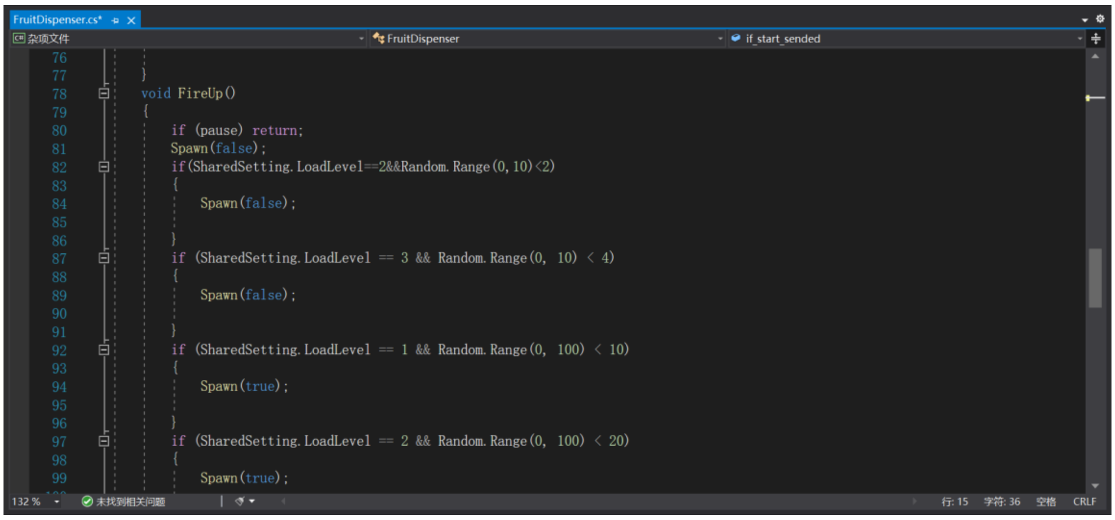


```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ObjectKill : MonoBehaviour
{
    // Start is called before the first frame update
    bool killed = false;
    public GameObject[] prefab;
    public float scale = 1f;
    public void OnKill()
    {
        if (killed)
        {
            return;
        }
        foreach (GameObject go in prefab)
        {
            GameObject ins = Instantiate(go, transform.position, Random.rotation) as GameObject;
            Rigidbody rd = ins.GetComponent<Rigidbody>();
            if (rd != null)
            {
                //rd.velocity = Random.onUnitSphere + Vector3.up;
                rd.velocity = Random.onUnitSphere + Vector3.up;
```

This script is FriutDispenser. First of all, we set different game difficulty levels in sharedsetting. In different difficulty levels, the fruit occurrence intervals are also different, namely 4s, 2s, 1.75s, 1.5s. The fireup () function also corresponds to the difficulty mentioned above.

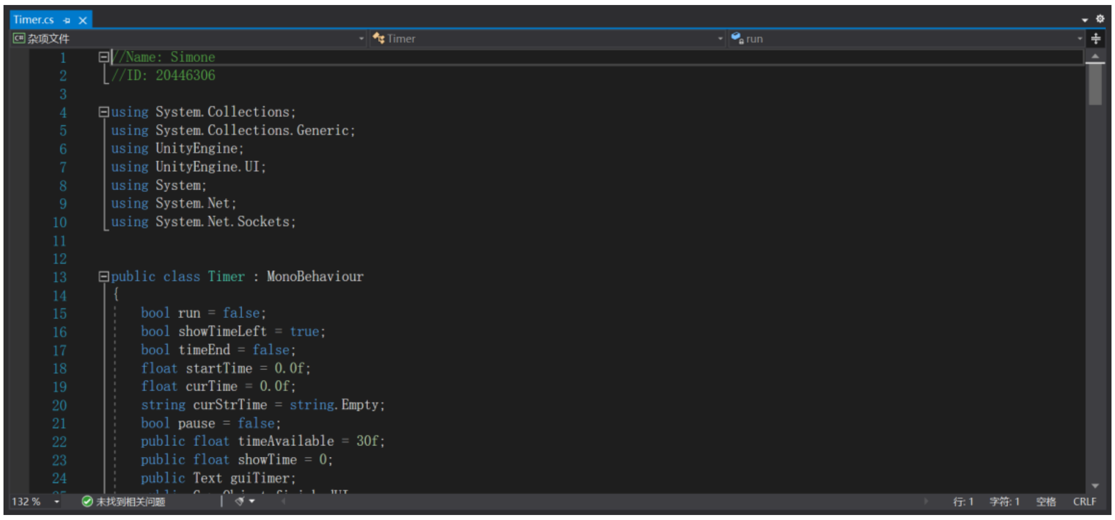


```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class FruitDispenser : MonoBehaviour
{

    public GameObject[] fruits;
    public GameObject bomb;
    public float y;
    public float powerScale;
    public bool pause = false;
    bool started = false;
    public float timer=2.5f;
    public int if_start_sended = 0;
    public bool is_first_start = false;

    public bool is_really_reieve;
    public bool is_really_reievel;
    // Start is called before the first frame update
    void Start()
    {

    }
```



```csharp
        }
    void FireUp()
    {
        if (pause) return;
        Spawn(false);
        if(SharedSetting.LoadLevel==2&&Random.Range(0,10)<2)
        {
            Spawn(false);

        }
        if (SharedSetting.LoadLevel == 3 && Random.Range(0, 10) < 4)
        {
            Spawn(false);

        }
        if (SharedSetting.LoadLevel == 1 && Random.Range(0, 100) < 10)
        {
            Spawn(true);

        }
        if (SharedSetting.LoadLevel == 2 && Random.Range(0, 100) < 20)
        {
            Spawn(true);
```

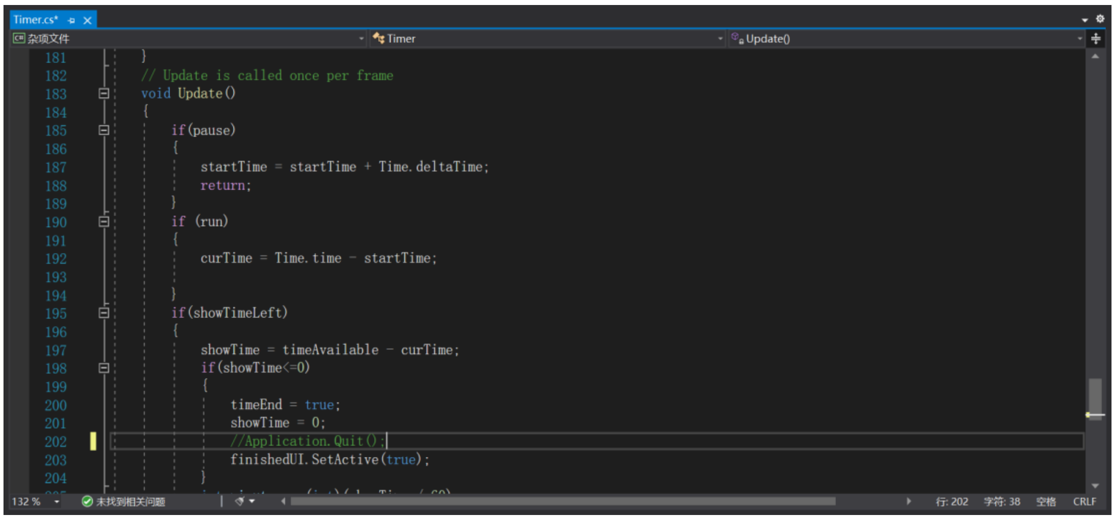
This is a script named Timer used to control the countdown. The purpose of this script is to calculate the remaining time according to the relationship between the total time and the current time. The variables are divided into hours, minutes and seconds. The current display time, namely the variable showtime, is the total time minus the initial time. Next, if the current time is less than or equal to 0, the UI component: Game End will be displayed on the screen. Then define minutes, seconds and fraction. The current time is calculated by the arithmetic operation of calculating minutes, hours and seconds. Finally, it is displayed on the screen in the format of UI elements in a fixed format.



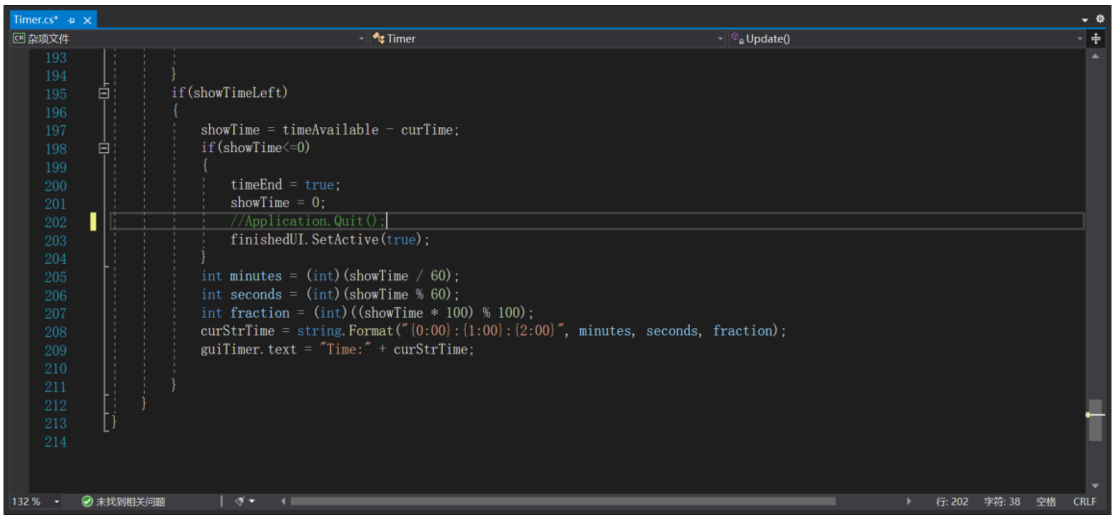
```csharp
//Name: Simone
//ID: 20446306

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


public class Timer : MonoBehaviour
{
    bool run = false;
    bool showTimeLeft = true;
    bool timeEnd = false;
    float startTime = 0.0f;
    float curTime = 0.0f;
    string curStrTime = string.Empty;
    bool pause = false;
    public float timeAvailable = 30f;
    public float showTime = 0;
    public Text guiTimer;
```

Let's take a look at this script. First, define the global variable in the global position to facilitate the reference of subsequent programs. The following notes are not related to the introduction of the game. This is another version that controls the motion of the hammer by sending signals from the server, but this time only introduces the version controlled by the VR handle.
Next, call the custom function RunTimer() in the built-in function start() in the C # script. The time to initialize the game is the current time when you click Start Game. Then continue the subsequent operations in the update() function. Note that the built-in update () function here is called every frame during the game running, so it can be regarded as a for loop with the frame number as the loop condition. Next, judge the current game state, pause and run. If the game is paused, the start time of the game will be accumulated continuously. If the game is running, subtract the game start time from the current time to get the current time displayed on the screen. Next, judge whether the

display time is 0. If it is less than or equal to 0, the game will end and prompt words will be displayed on the screen. Then calculate the time displayed on the screen, use division to calculate minutes, use remainder to calculate seconds, and use multiplication and remainder to calculate milliseconds. Finally, it is displayed on the UI components in the screen in a fixed form.

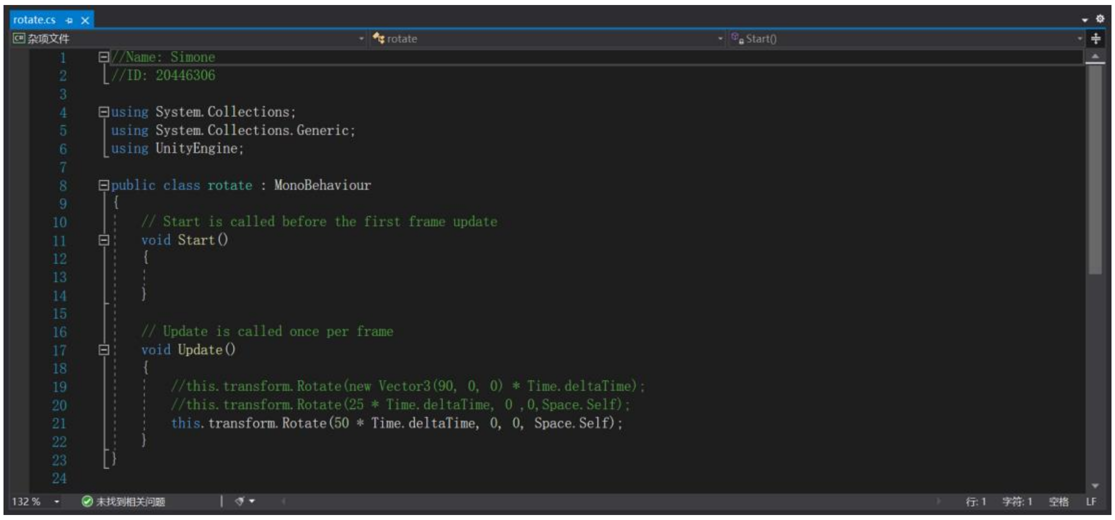


```
    }
    // Update is called once per frame
    void Update()
    {
        if(pause)
        {
            startTime = startTime + Time.deltaTime;
            return;
        }
        if (run)
        {
            curTime = Time.time - startTime;

        }
        if(showTimeLeft)
        {
            showTime = timeAvailable - curTime;
            if(showTime<=0)
            {
                timeEnd = true;
                showTime = 0;
                //Application.Quit();
                finishedUI.SetActive(true);
            }
```



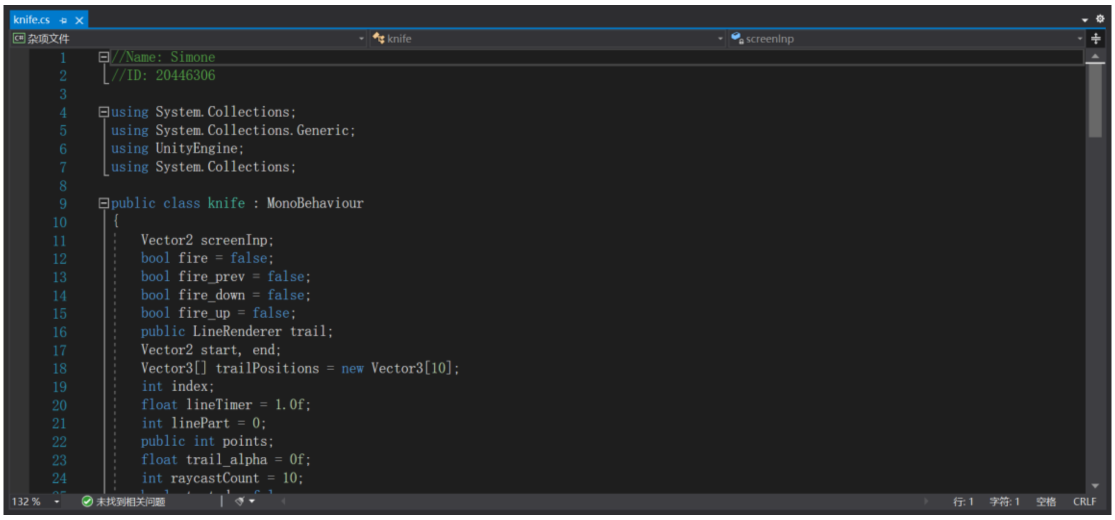
```
        }
        if(showTimeLeft)
        {
            showTime = timeAvailable - curTime;
            if(showTime<=0)
            {
                timeEnd = true;
                showTime = 0;
                //Application.Quit();
                finishedUI.SetActive(true);
            }
            int minutes = (int)(showTime / 60);
            int seconds = (int)(showTime % 60);
            int fraction = (int)((showTime * 100) % 100);
            curStrTime = string.Format("{0:00}:{1:00}:{2:00}", minutes, seconds, fraction);
            guiTimer.text = "Time:" + curStrTime;
        }
    }
}
```

This script which called rotate is used to control the rolling action during fruit movement. This script is relatively simple, because it only implements an object rolling action by calling the transform. Rotate function. transform. The parameters in the Rotate function are used to control the x axis rolling direction, y axis rolling direction, z axis rolling direction and the reference of the three coordinate axes respectively. This statement is placed in the update() function and is executed continuously with each frame call.



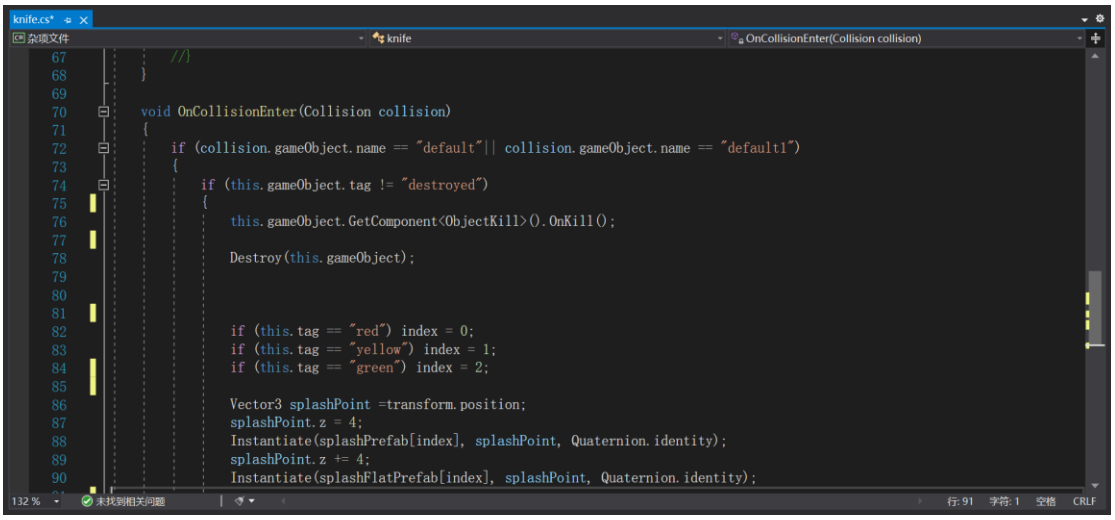
```csharp
//Name: Simone
//ID: 20446306

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

public class rotate : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        //this.transform.Rotate(new Vector3(90, 0, 0) * Time.deltaTime);
        //this.transform.Rotate(25 * Time.deltaTime, 0 ,0,Space.Self);
        this.transform.Rotate(50 * Time.deltaTime, 0, 0, Space.Self);
    }
}
```

Next is the script which is named kinfe. First judge whether the name of the collision body is the name of two hammers. If yes, continue to execute the part that breaks the fruit. First, initialize the cut part of the fruit to form an array, then delete the original complete fruit, mark the different split parts of the three kinds of fruits with different tags, that is, tags, and finally generate the liquid generated when the fruit is cut, that is, a flash component.



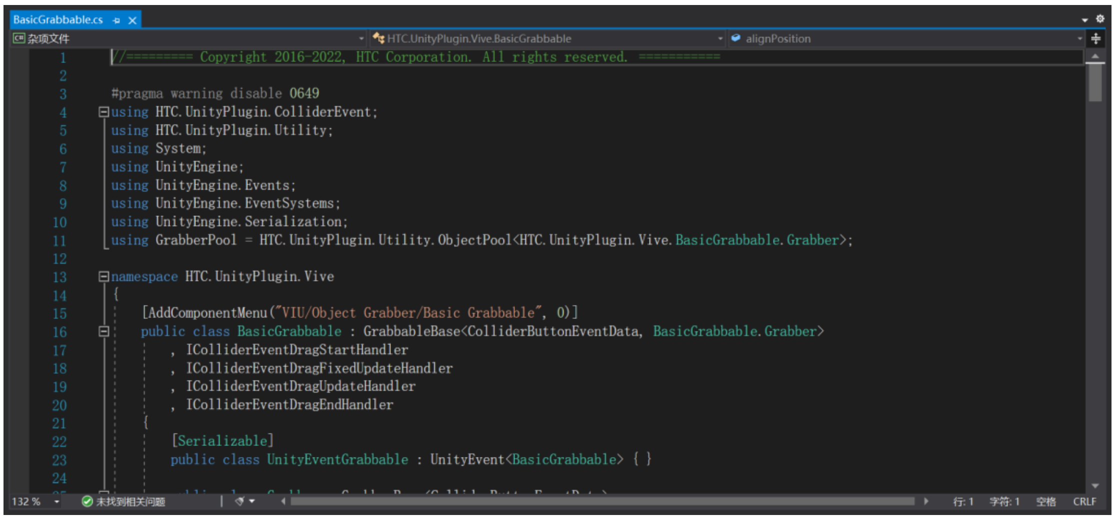
```csharp
//Name: Simone
//ID: 20446306

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

public class knife : MonoBehaviour
{
    Vector2 screenInp;
    bool fire = false;
    bool fire_prev = false;
    bool fire_down = false;
    bool fire_up = false;
    public LineRenderer trail;
    Vector2 start, end;
    Vector3[] trailPositions = new Vector3[10];
    int index;
    float lineTimer = 1.0f;
    int linePart = 0;
    public int points;
    float trail_alpha = 0f;
    int raycastCount = 10;
```

Let's take a look at this script. First, define global variables at the beginning for easy use. This script does not use the c # initial functions start() and update(), but instead writes a new function OnColisionEnter() to detect Rigidbody and box collider, judge whether the collision body is the two hammers in the game scene through the if statement, and continue the next procedure. Next, if the fruit successfully collides with the hammer, the cut fruit will be generated in the scene, and the original complete fruit will be deleted without time interval. Next, mark the cut fruits and mark them with different tag.



```
knife.cs*
杂项文件                    knife                    OnCollisionEnter(Collision collision)
            //}
        }

        void OnCollisionEnter(Collision collision)
        {
            if (collision.gameObject.name == "default"|| collision.gameObject.name == "default1")
            {
                if (this.gameObject.tag != "destroyed")
                {
                    this.gameObject.GetComponent<ObjectKill>().OnKill();

                    Destroy(this.gameObject);



                    if (this.tag == "red") index = 0;
                    if (this.tag == "yellow") index = 1;
                    if (this.tag == "green") index = 2;

                    Vector3 splashPoint =transform.position;
                    splashPoint.z = 4;
                    Instantiate(splashPrefab[index], splashPoint, Quaternion.identity);
                    splashPoint.z += 4;
                    Instantiate(splashFlatPrefab[index], splashPoint, Quaternion.identity);
132 %    未找到相关问题                                          行: 91   字符: 1   空格   CRLF
```

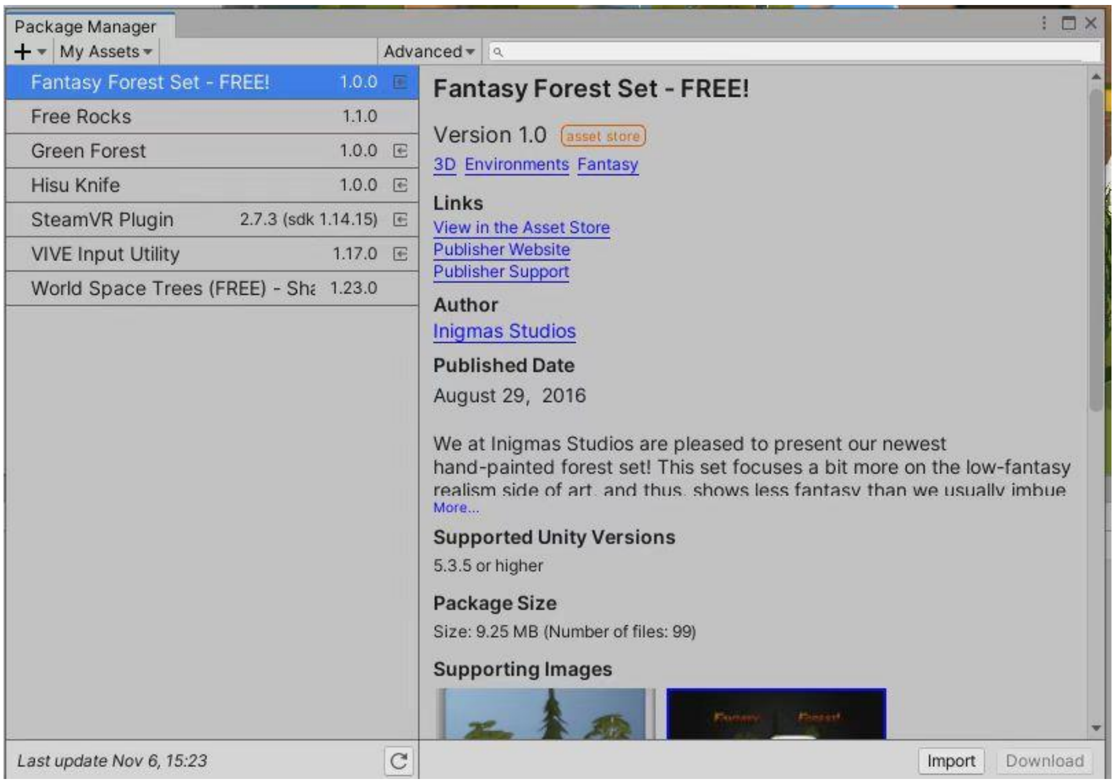
This is a necessary script named BasicGrabbable that comes with the connection between VR devices and unity, and is attached to objects captured by VR handles. The specific content of the script will not be detailed here



```
BasicGrabbable.cs
杂项文件                    HTC.UnityPlugin.Vive.BasicGrabbable                    alignPosition
//========= Copyright 2016-2022, HTC Corporation. All rights reserved. ===========

#pragma warning disable 0649
using HTC.UnityPlugin.ColliderEvent;
using HTC.UnityPlugin.Utility;
using System;
using UnityEngine;
using UnityEngine.Events;
using UnityEngine.EventSystems;
using UnityEngine.Serialization;
using GrabberPool = HTC.UnityPlugin.Utility.ObjectPool<HTC.UnityPlugin.Vive.BasicGrabbable.Grabber>;

namespace HTC.UnityPlugin.Vive
{
    [AddComponentMenu("VIU/Object Grabber/Basic Grabbable", 0)]
    public class BasicGrabbable : GrabbableBase<ColliderButtonEventData, BasicGrabbable.Grabber>
        , IColliderEventDragStartHandler
        , IColliderEventDragFixedUpdateHandler
        , IColliderEventDragUpdateHandler
        , IColliderEventDragEndHandler
    {
        [Serializable]
        public class UnityEventGrabbable : UnityEvent<BasicGrabbable> { }

132 %    未找到相关问题                                          行: 1   字符: 1   空格   CRLF
```

3. VR Connection

About how unity games connect VR devices. First, download the control script files related to the VR device, live, in the unity store.


Package Manager

My Assets | Advanced

Fantasy Forest Set - FREE!	1.0.0
Free Rocks	1.1.0
Green Forest	1.0.0
Hisu Knife	1.0.0
SteamVR Plugin	2.7.3 (sdk 1.14.15)
VIVE Input Utility	1.17.0
World Space Trees (FREE) - Shε	1.23.0

Last update Nov 6, 15:23

Fantasy Forest Set - FREE!

Version 1.0 asset store

3D Environments Fantasy

Links

View in the Asset Store

Publisher Website

Publisher Support

Author

Inigmas Studios

Published Date

August 29, 2016

We at Inigmas Studios are pleased to present our newest hand-painted forest set! This set focuses a bit more on the low-fantasy realism side of art, and thus, shows less fantasy than we usually imbue

More...

Supported Unity Versions

5.3.5 or higher

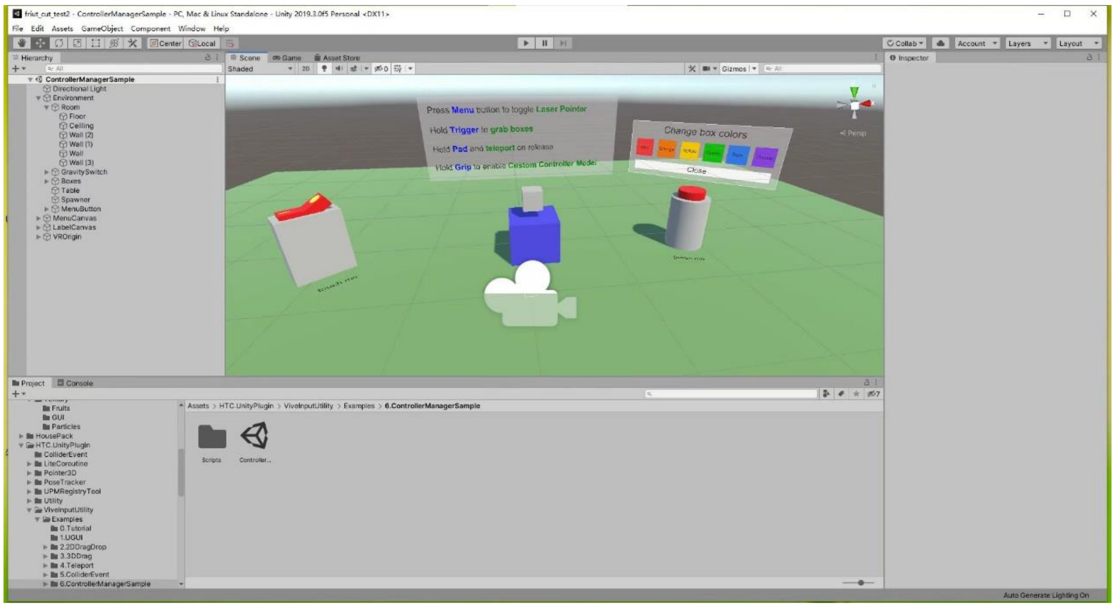
Package Size

Size: 9.25 MB (Number of files: 99)

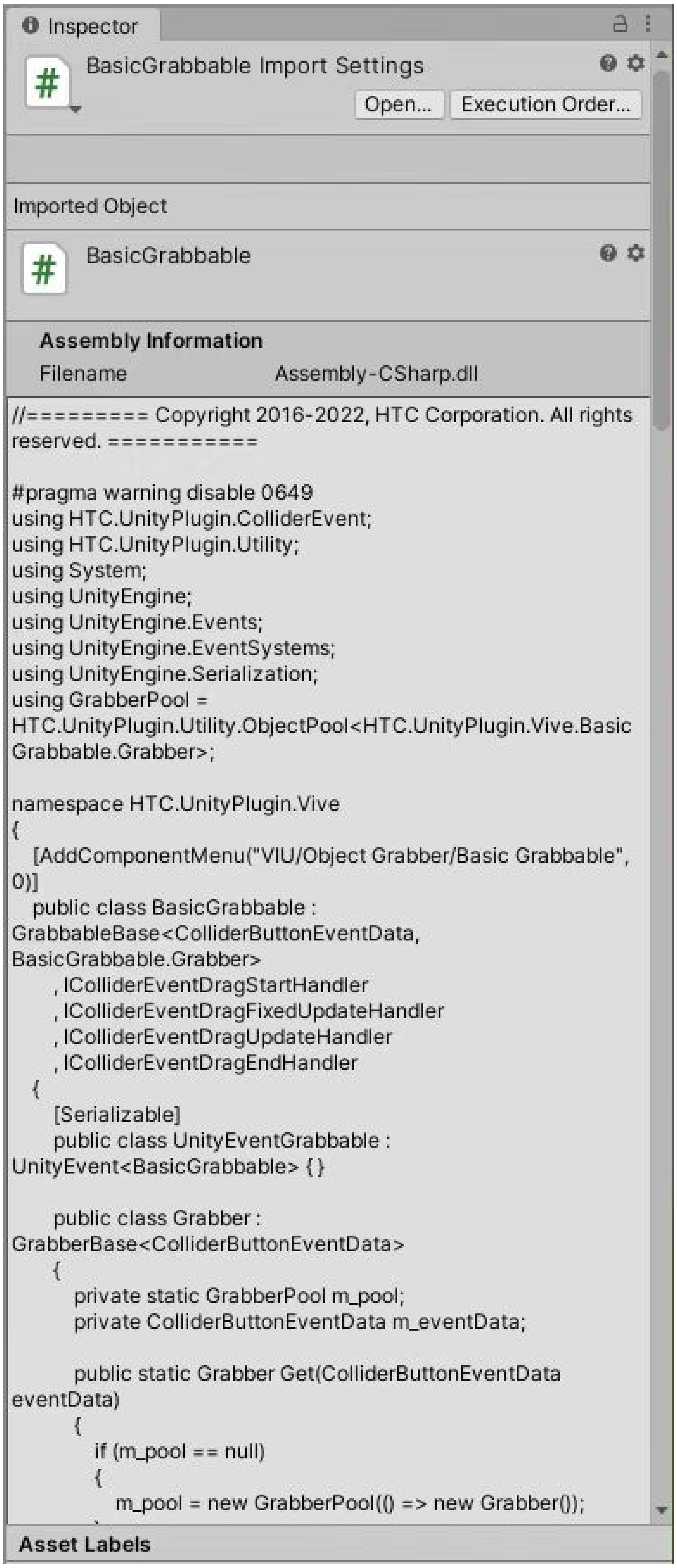
Supporting Images

Import Download

After opening the file, first refer to the Demo Sense that comes with the file. After trying to run the Demo Sense, understand the general idea of the whole VR device running in unity.

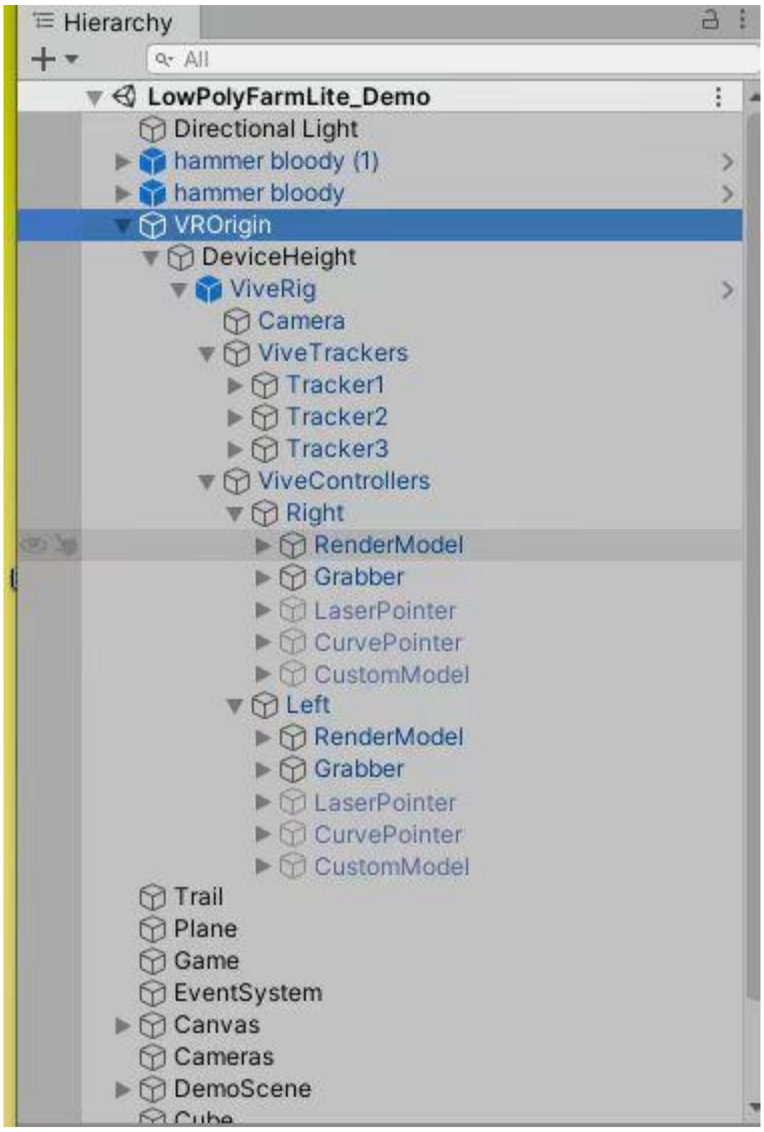

Hierarchy
ControllerManagerSample
Scene
Game
Asset Store
Inspector
Press Menu button to toggle Laser Pointer
Hold Trigger to grab boxes
Hold Pad and teleport on release
Hold Grip to enable Custom Controller Mode
Change box colors
Close
Project
Console
Assets > HTC.UnityPlugin > ViveInputUtility > Examples > 6.ControllerManagerSample

Then open our own unity game for connecting VR, attach the capture script to the object that needs to be captured by the handle, and modify Rigidbody's collision logic so that the object that can be captured can collide with other objects in the VR connection mode.



```
//========= Copyright 2016-2022, HTC Corporation. All rights
reserved. ===========

#pragma warning disable 0649
using HTC.UnityPlugin.ColliderEvent;
using HTC.UnityPlugin.Utility;
using System;
using UnityEngine;
using UnityEngine.Events;
using UnityEngine.EventSystems;
using UnityEngine.Serialization;
using GrabberPool =
HTC.UnityPlugin.Utility.ObjectPool<HTC.UnityPlugin.Vive.Basic
Grabbable.Grabber>;

namespace HTC.UnityPlugin.Vive
{
  [AddComponentMenu("VIU/Object Grabber/Basic Grabbable",
0)]
  public class BasicGrabbable :
GrabbableBase<ColliderButtonEventData,
BasicGrabbable.Grabber>
      , IColliderEventDragStartHandler
      , IColliderEventDragFixedUpdateHandler
      , IColliderEventDragUpdateHandler
      , IColliderEventDragEndHandler
  {
      [Serializable]
      public class UnityEventGrabbable :
UnityEvent<BasicGrabbable> { }

      public class Grabber :
GrabberBase<ColliderButtonEventData>
      {
         private static GrabberPool m_pool;
         private ColliderButtonEventData m_eventData;

         public static Grabber Get(ColliderButtonEventData
eventData)
         {
            if (m_pool == null)
            {
               m_pool = new GrabberPool(() => new Grabber());
```

At the same time, it is also necessary to import another component into our game, namely VROrigin. After importing this component, the physical location of the handle can be displayed when the game is running, so that players can locate the position of the handle and use the handle to grab objects.



```
Hierarchy
+ ▾  All
▼ LowPolyFarmLite_Demo
    Directional Light
  ▶ hammer bloody (1)
  ▶ hammer bloody
  ▼ VROrigin
    ▼ DeviceHeight
      ▼ ViveRig
          Camera
        ▼ ViveTrackers
          ▶ Tracker1
          ▶ Tracker2
          ▶ Tracker3
        ▼ ViveControllers
          ▼ Right
            ▶ RenderModel
            ▶ Grabber
            ▶ LaserPointer
            ▶ CurvePointer
            ▶ CustomModel
          ▼ Left
            ▶ RenderModel
            ▶ Grabber
            ▶ LaserPointer
            ▶ CurvePointer
            ▶ CustomModel
    Trail
    Plane
    Game
    EventSystem
  ▶ Canvas
    Cameras
  ▶ DemoScene
    Cube
```

The whole game process does not need to be modified after connecting to the VR device. Games that can run in unity can still run after connecting to the VR device, and run from the perspective of VR glasses, that is, panoramic view. If the above scripts and components are not attached, the game can only be viewed from the VR perspective and cannot be controlled.

4. Data Collection

(1) Physical disability data in China

There are 650 million people with disabilities worldwide, accounting for about 10% of the world's total population, of which 80% are distributed in developing countries.

The results of the second national sample survey of the disabled people show that among the disabled people in China, the largest number of physical disabilities, with 29.77 million, accounting for 29.07% of the total number of disabled people. Of these physical disabilities, 27.69% had a need for rehabilitation training, while only 8.45% had received and were receiving rehabilitation.

In many countries, up to 80% of people with disabilities are unable to find jobs after reaching employment age, which is much higher than normal people.
Therefore, it is very important to help the physically disabled to recover.
Rehabilitation can help the disabled recovery or compensation function, improve the quality of life, enhance the ability of social participation, is the disabled to school, employment, poverty, equal participation in social life, realize the premise of well-off, through rehabilitation can help the disabled stand up, see the world, speak, help them regain the right to life, start a new life.
After the successful operation of the game, collect and analyze the data of the experiencer to obtain better experience results.

(2) The benefits of unity Engine

a. Intelligent interface design, visual programming interface to complete a variety of development work, efficient script editing, to achieve efficient development
b. Component-oriented development, different game projects, the whole logic needs to be rewritten in C#, but the resources can be used in the original project.
c. Cross-platform, multi-platform development and deployment of the work can be completed with one key, supporting PC, mobile and host platforms
d. easy to get started, the development language C# is easier than C/C++, and the use of unity has been relatively mature, more learning videos
e. Support ARVR. U3D is the most mainstream development engine to achieve ARVR

(3) The advanced nature of VR games

VR games can make people move, which can help patients recover. The immersive experience makes the player feel more engaged and interactive.
a. The faker-like existence technology syntheses the input information consistent with the actual existence for the sensory organs of the observer, which can produce the same vision, touch, smell and so on as the real environment.
b. Interaction. Observers actively and actively operate virtual reality to achieve different view scenes and higher levels of sensory information.
c. Disciplined reality. The perceiver acquires a vivid sense of reality without being aware of his actions or actions. Here, the observer, the sensor, the computer simulation system and the display system constitute a closed-loop process of interaction.

(4) The advanced nature of unity in mainly using c # to write games

First, C# has a more advanced syntax than Java, and many of the syntax in C# is an improvement on other languages. As a commercial product, C++ users are the primary target, so it takes care of C++ user habits. Overall, C# syntax is more elegant than java. The C# team designed the syntax with common business requirements in mind, rather than purely technical ideas. As a result, C# supports events, delegates, attributes, Linq, and a host of other features that make business development easier.
Second, C# has powerful peripherals. C# IDE is very powerful, C# documentation is available in multiple languages including Chinese, and C# needs to run on a platform built into windows with a large user base. Everything tells us that C# has a powerful father. It also shows that C# is not just a language, but a product of great ingenuity. Therefore, users can enjoy all the functions of this product ---- including after-sales service.
C# has a modern, versatile programming language, it can produce efficient programs, and it can be compiled on a variety of computer platforms. It has a lot of powerful programming features,

which is suitable for our unity game.

5. Team Members

Annie 20446305
Simone 20446306
Luke   20440325
nini 20446404
Siri 20416304
Nicole 20416305
Cloud 20435523