G-Wearables

STEPVR SDK 使用说明(C#)

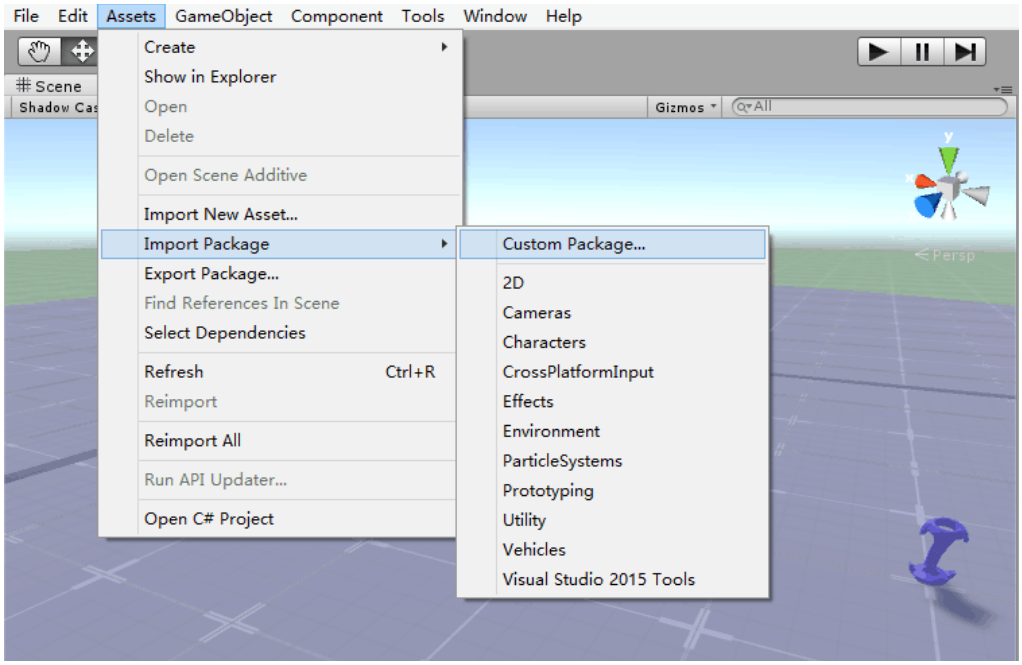
1 运行环境

1.1 硬件环境: StepVR 标准件, StepVR 手柄, OculusCV1 头显

1.2 软件环境: Unity5.3.4 及以上

2 SDK 导入

在 Unity 中导入 SDK，点击 Assets->Import Package->Custom Package...，选择 SDKforUnity



3 SDK 使用

3.1 引用命名空间：using StepVRSDKUnity;

3.2 数据类型:

```
struct V3 {
    public float x;
    public float y;
    public float z;
 }
```

说明:声明三元数的数据类型

```
struct V4 {
    public float x;
    public float y;
    public float z;
    public float w;
}
```

说明:声明四元数的数据类型

```
public enum EquipmentNode {
        Head = 6,
        Hand = 4,
        LeftController = 1,
        RightController = 2
}
```

说明:设备类别

3.3 实例化 StepVRManager

实例化 StepVRManager 类后可调用获取 stepVR 硬件数据的方法.(注:只能实例化一次)

3.4 全部方法

```
void StartSDK()
说明:  启动 stepVRsdk;
(注:建议在 Awake()或 Start()中调用)
```

```
void StopSDK()
说明:停用 stepVRsdk;
(注:建议在 OnDestroy()中调用)
```

V3 GetPosition(EquipmentNode node)

说明:获取设备位置数据.

参数:设备类别

返回值:V3 类型的 x,y,z 轴坐标

V3 GetRotation(EquipmentNode node)

说明:获取设备姿态数据.

参数:设备类别

返回值:V3 类型的 x,y,z 轴坐标

V4 GetQuaternion(EquipmentNode node)

说明:获取设备姿态数据.

参数:设备类别

返回值:V4 类型的 x,y,z,w 轴坐标

bool GetTriggerDown(EquipmentNode node, string key)

说明:获取按键按下状态(按下)

参数:设备类别,按键名(A,B,C)

返回值:是否按下,按下为 true,未按下为 false

bool GetTrigger(EquipmentNode node, string key)

说明:获取按键按住状态(长按)

参数:设备类别,按键名(A,B,C)

返回值:是否按下,按下为 true,未按下为 false

bool GetTriggerUp(EquipmentNode node, string key);

说明:获取按键抬起状态(抬起)

参数:设备类别,按键名(A,B,C)

返回值:是否抬起,抬起为 true,未抬起为 false

4 示例程序：

4.1 写一个单独脚本 StepVRSingleton.cs，包含如下：

```
//设备的ID
public EquipmentNode gunID;//枪的ID
public EquipmentNode headID;//头的ID
public EquipmentNode leftControllerID;//左手手柄的ID
public EquipmentNode rightControllerID;//右手手柄的ID

//两个手柄的按键信息
public bool handAKeyADown;
public bool handAKeyBDown;
public bool handAKeyCDown;

public bool handBKeyADown;
public bool handBKeyBDown;
public bool handBKeyCDown;

//小绿人中是否有手柄
public bool isHand;
```

```
//实例化一个StepVRManager
StepVRManager manager = new StepVRManager();
2 个引用
public bool Start()
{
    if (manager == null)
    {
        return false;
    }
    else
    {
        manager.StartSDK();
        return true;
    }
```

```
2 个引用
public Vector3 EquipmentPosition(EquipmentNode node)
{
    return new Vector3(manager.GetPosition(node).x, manager.GetPosition(node).y, manager.GetPosition(node).z);
}

2 个引用
public Quaternion EquipmentRotation(EquipmentNode node)
{
    return Quaternion.Euler(new Vector3(manager.GetRotation(node).x, manager.GetRotation(node).y, manager.GetRotation(node).z));
}

2 个引用
public bool Trigger(EquipmentNode node, string key)
{
    return manager.GetTrigger(node, key);
}

2 个引用
public bool TriggerDown(EquipmentNode node, string key)
{
    return manager.GetTriggerDown(node, key);
}

0 个引用
public bool TriggerUp(EquipmentNode node, string key)
{
    return manager.GetTriggerUp(node, key);
}
```

4.2 再写一个 palyer.cs 脚本，绑定在人物上

```
0 个引用
void Start()
{
    SetRoomSize(StepVRSingleton.Instance.x, StepVRSingleton.Instance.z);
    if (!isSDKStart)
    {
        //调用SDK,启用StepVRsdk,StartSDK()函数再Start()函数内
        isSDKStart = StepVRSingleton.Instance.Start();
    }
```

```
    Vector3 headPosition = GetNodePosition(StepVRSingleton.Instance.headID);
    //输出头部数据，头部定位件的位置信息
    txtHead.text = "头:\nx:" + headPosition.x + "\ny:" + headPosition.y + "\nz:" + headPosition.z;
    //transform.position = headPosition;
    transform.position = new Vector3(headPosition.x, 0, headPosition.z);
    headJoint.localPosition = new Vector3(0, headPosition.y, 0);

    //CheckIfOutOfBound(headPosition.x, headPosition.z);
    //输出枪的数据，枪上定位件的位置信息
    gunJoint.position = GetNodePosition(StepVRSingleton.Instance.gunID);
    txtGun.text = "枪位移:\nx:" + gunJoint.position.x + "\ny:" + gunJoint.position.y + "\nz:" + gunJoint.position.z;
    //枪上定位件的姿态信息
    gunJoint.rotation = GetNodeRotation(StepVRSingleton.Instance.gunID);
    txtGunRot.text = "枪姿态:\nx:" + gunJoint.rotation.x + "\ny:" + gunJoint.rotation.y + "\nz:" + gunJoint.rotation.z;
    UpdateText();
    CastRay();
```

```
        //开火，枪的按键检测
        if (GetKeyDown(StepVRSingleton.Instance.gunID, "A") || GetKey(StepVRSingleton.Instance.gunID, "A")
            || Input.GetMouseButton(1))
        {
            if (guns[currenGunIndex].isActiveAndEnabled)
                guns[currenGunIndex].Shoot();
            if (currentPointingButton)
                currentPointingButton.ToggleButton();
        }

        //获取手柄信息
        if (leftController.activeSelf && rightController.activeSelf)
        {
            handA.position = GetNodePosition(StepVRSingleton.Instance.leftControllerID);//手柄A的位置
            handA.rotation = GetNodeRotation(StepVRSingleton.Instance.leftControllerID);//手柄A的姿态
            handB.position = GetNodePosition(StepVRSingleton.Instance.rightControllerID);//手柄B的位置
            handB.rotation = GetNodeRotation(StepVRSingleton.Instance.rightControllerID);//手柄B的姿态
```

```
            //判断按键
            if (GetKeyDown(StepVRSingleton.Instance.leftControllerID, "A")
                || GetKey(StepVRSingleton.Instance.leftControllerID, "A"))
            {
                txtTriggerA.text = "handA按键信息:A";
                Debug.Log("handA按键信息:A");
                StepVRSingleton.Instance.handAKeyADown = true;
            }
            if (GetKeyDown(StepVRSingleton.Instance.leftControllerID, "B")
                || GetKey(StepVRSingleton.Instance.leftControllerID, "B"))
            {
                txtTriggerB.text = "handA按键信息:B";
                Debug.Log("handA按键信息:B");
                StepVRSingleton.Instance.handAKeyBDown = true;
            }
            if (GetKeyDown(StepVRSingleton.Instance.leftControllerID, "C")
                || GetKey(StepVRSingleton.Instance.leftControllerID, "C"))
            {
                txtTriggerC.text = "handA按键信息:C";
                Debug.Log("handA按键信息:C");
                StepVRSingleton.Instance.handAKeyCDown = true;
                Instantiate(ballA, handA.position, Quaternion.identity);//按住则实例化小球
            }
```

```
    0 个引用
    void OnDestroy()
    {
        //关闭SDK，停用StepVRsdk,StopSDK()函数在Stop()函数内
        StepVRSingleton.Instance.Stop();
    }
```

```
    //获取定位点位置数据，使用GetPosition(EquipmentNode node)
    4 个引用
    Vector3 GetNodePosition(EquipmentNode node)
    {
        return StepVRSingleton.Instance.EquipmentPosition(node);
    }


    //获取定位点姿态数据，使用EquipmentRotation(EquipmentNode node)
    3 个引用
    Quaternion GetNodeRotation(EquipmentNode nodeIndex)
    {
        return StepVRSingleton.Instance.EquipmentRotation(nodeIndex);
    }


    //获取手柄按键状态，使用Trigger(EquipmentNode node, string key)
    7 个引用
    bool GetKey(EquipmentNode node, string key)
    {
        return StepVRSingleton.Instance.Trigger(node, key);
    }


    //获取手柄按键按下状态，使用TriggerDown(EquipmentNode node, string key)
    7 个引用
    bool GetKeyDown(EquipmentNode node, string key)
    {
        return StepVRSingleton.Instance.TriggerDown(node, key);
    }

    /*
    //获取手柄按键抬起状态，使用TriggerUp(EquipmentNode node, string key)
    bool GetKeyUp(EquipmentNode node, string key)
    {
```