

Hi5 2.0 Pico Unity 交互 SDK 使用文档

1.确保每个平台 Hi5_2 SDK 使用正常

2.各平台分别导入交互式 SDK package

Steam VR 平台导入 Hi5_2_Interaction_Vive.unitypackage。

Pico Neo 3 平台导入 Hi5_2_Interaction_Pico.unitypackage。

HTC Vive Focus 平台导入 Hi5_2_Interaction_HTCViveFocus.unitypackage。

3.设置 Layer 和 Physics (必须设置)

Layer 8 Hi5OtherFingerTail

Layer 9 Hi5OtherFingerOther

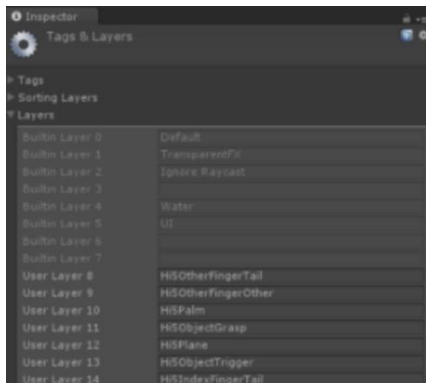
Layer 10 Hi5Palm

Layer 11 Hi5ObjectGrasp

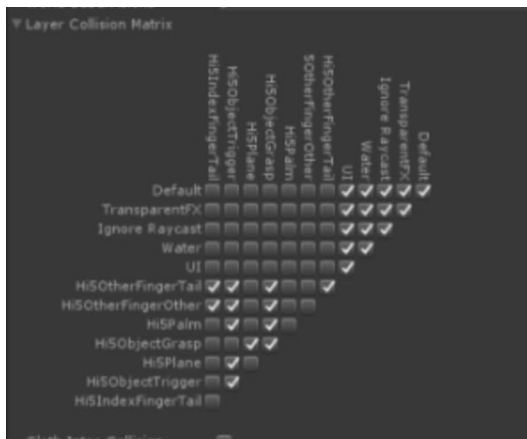
Layer 12 Hi5Plane

Layer 13 Hi5ObjectTrigger

Layer 14 Hi5IndexFingerTail



Edit->Project Setting->Physics



4.演示场景

Pico 平台 打开 TableScene_Pico

Steam VR 平台 打开 TableScene_Vive

HTC Vive Focus 平台 打开 TableScene_ViveFocus (手柄驱动)、
TableScene_ViveFocusWrist (手环驱动)

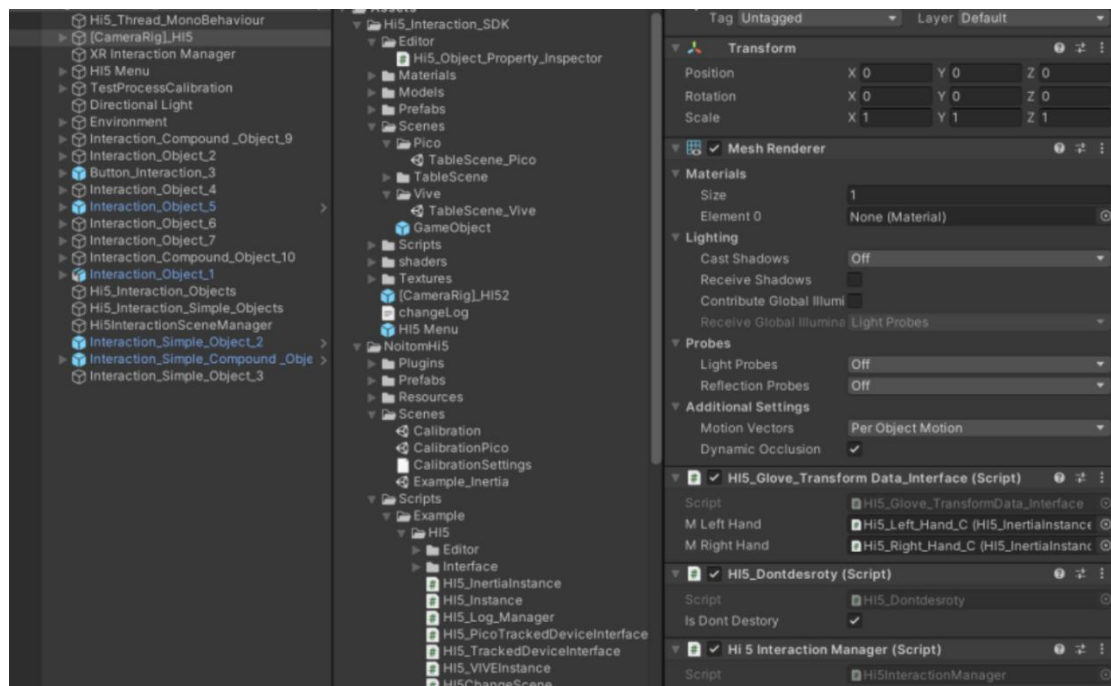
YVR 平台 打开 TableScene_YVR

初始包设置完成就可以运行示例场景运行。

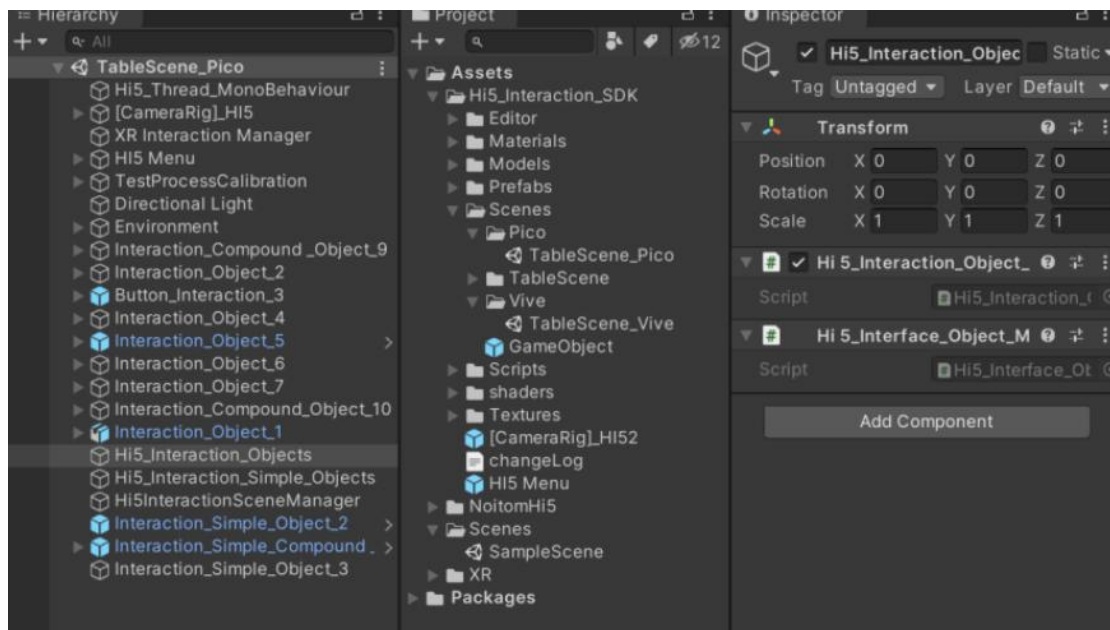
5.使用方法

5.1 交互场景必须有以下内容

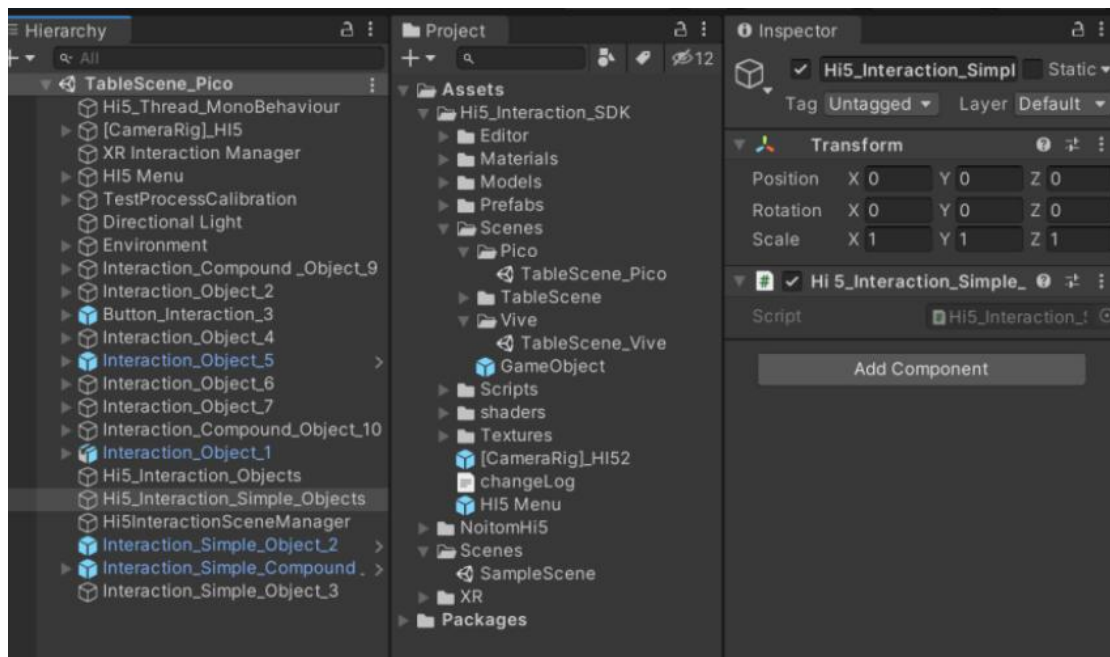
5.1.1.Hi5InteractionManager



5.1.2.Hi5_Interaction_Objects



5.1.3.Hi5_Interaction_Simple_Objects



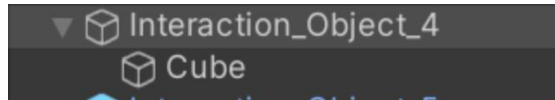
5.1.4 Hi5_Left_Hand_C Hi5_Right_Hand_C Hi5_Left_Hand_V Hi5_Right_Hand_V



5.2 场景物体设置

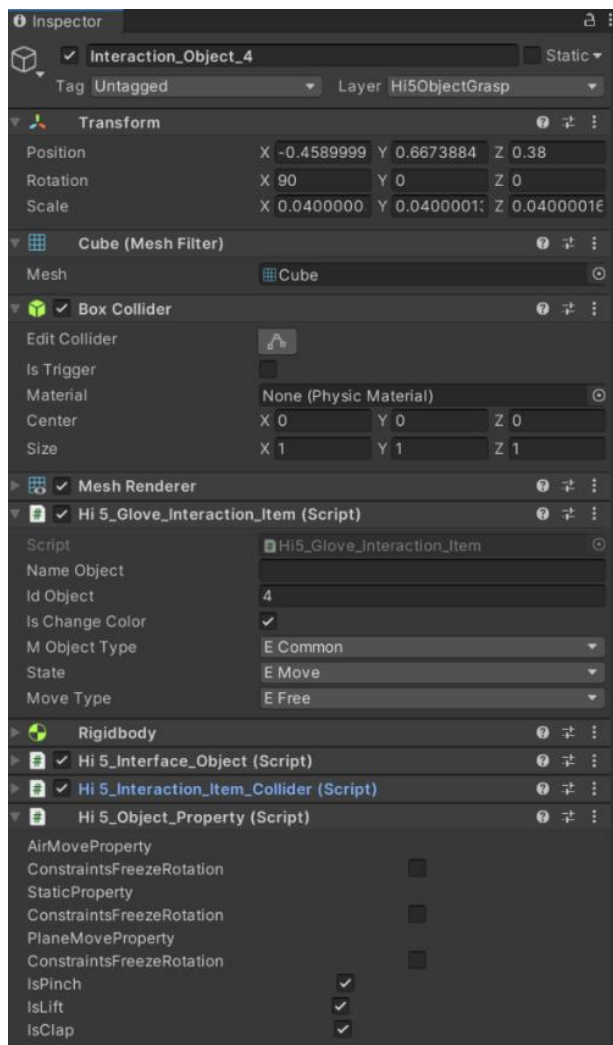
5.2.1.普通交互物体设置

物体设置分物体本身及子物体



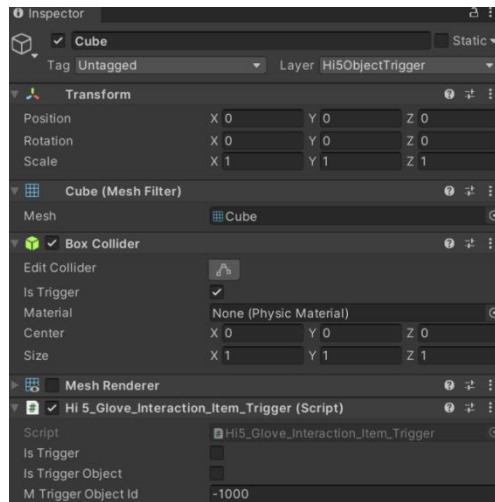
主物体设置

注意 Layer 设置为 Hi5ObjectGrasp

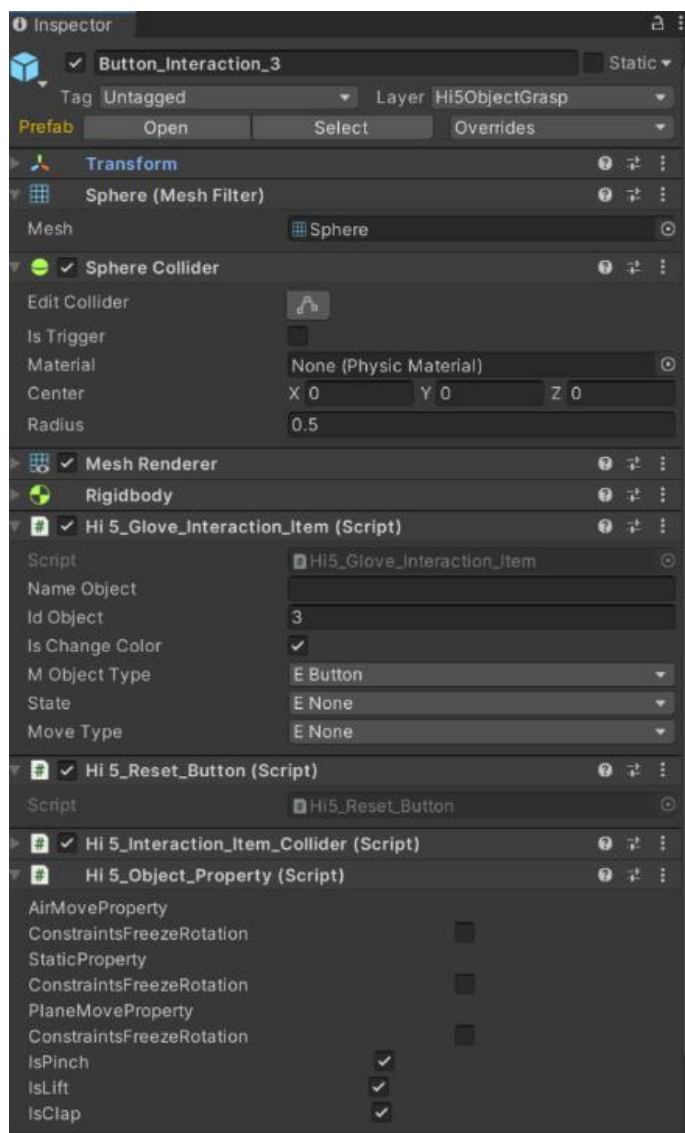


子物体设置

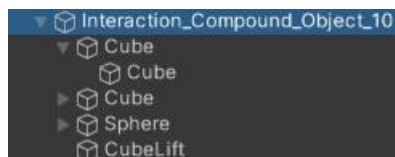
注意 layer 层设置为 Hi5ObejctTrigger



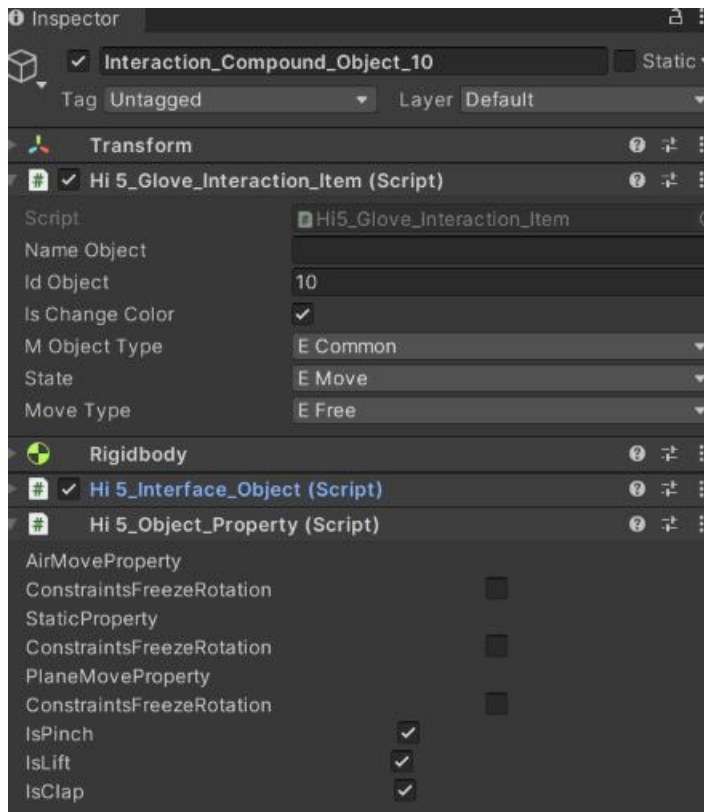
5.2.2.button 设置



5.2.3.组合物体设置



组合物体分为三层

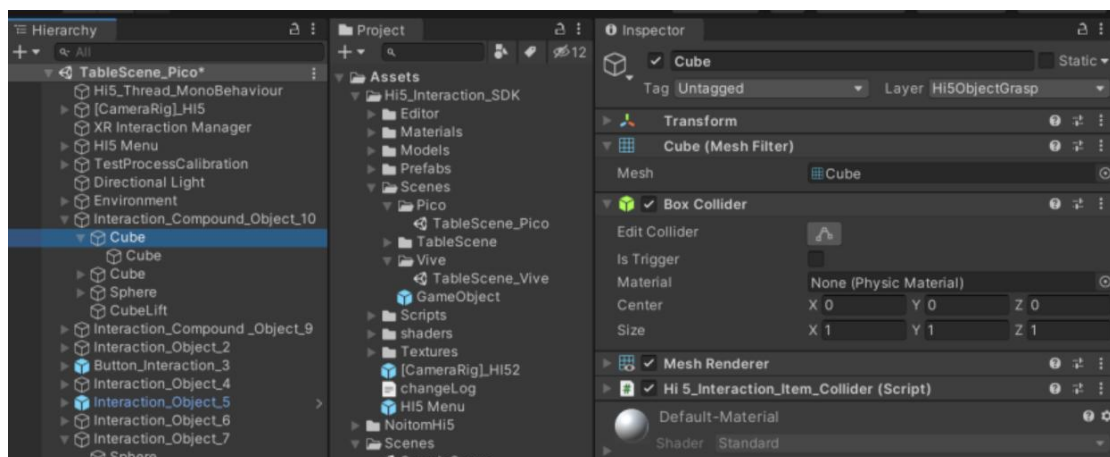


物体最外层物体本身设置

子组合体设置

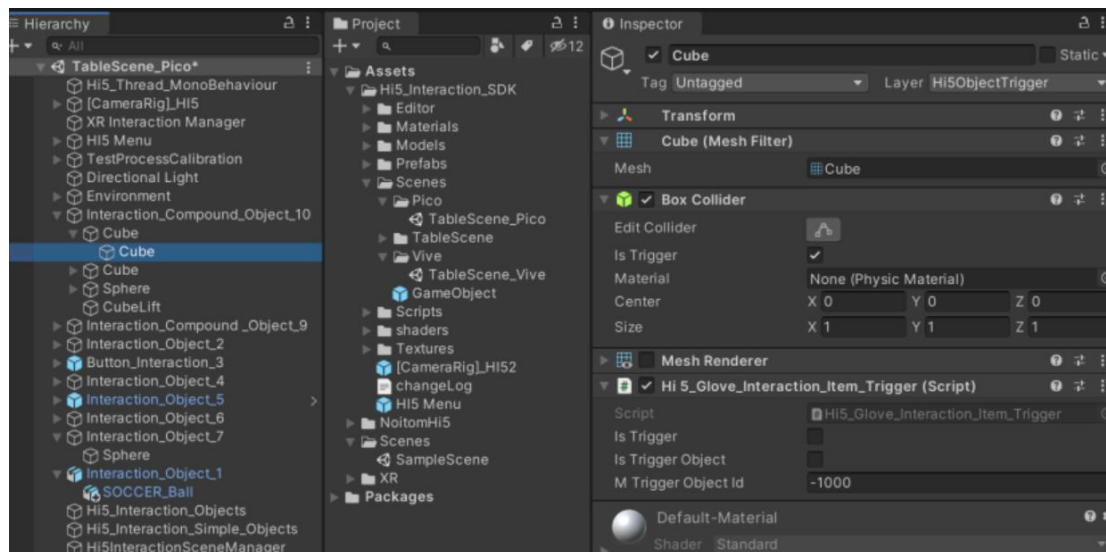
子组合体外层设置

注意 Layer 设置为 Hi5ObjectGrasp



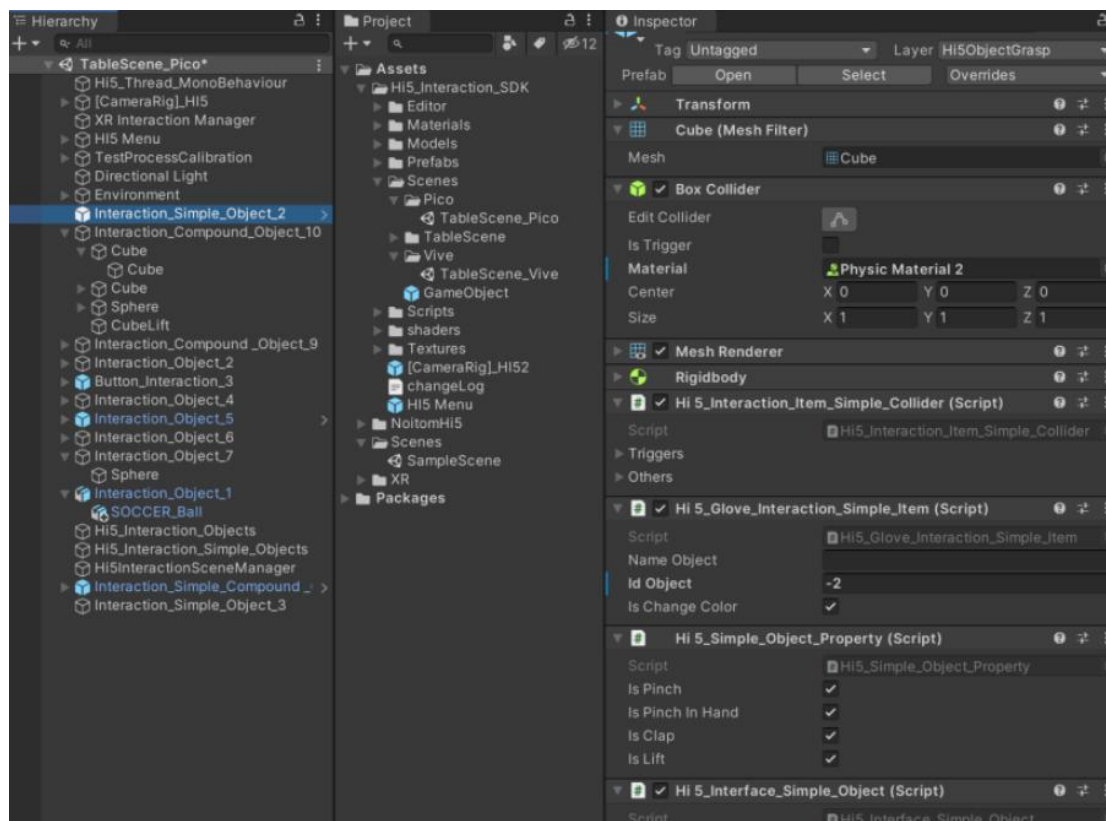
子组合体内层设置

注意 layer 层设置为 Hi5ObjectTrigger

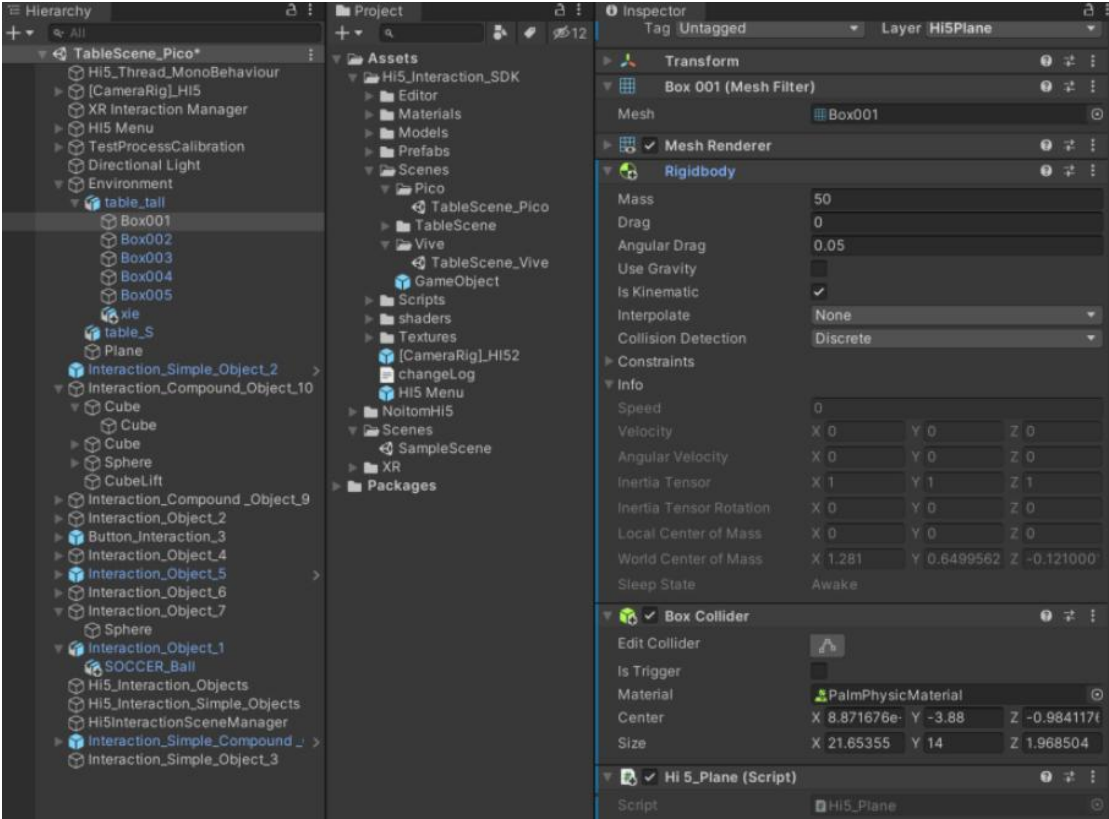


4.简单物体设置 简单物体只有抓握等功能，自身不会产生运动，当抓住释放后会停留在原位置。

注意 Layer 设置为 Hi5ObjectGrasp



5.3 桌面设置场景中可以放置物体地方需要设置 Hi5_Plane 脚本 例如地面桌子等



6.相关接口

6.1.手相关接口

Hi5_Interface_Hand 脚本

6.1.1 手状态

enum E_Interface_Hand_State

{

ERelease = -1,

EPinch = 2,

ELift = 4,

}

E_Interface_Hand_State GetHandState(out int interactionObjectId)

E_Interface_Hand_State 返回手部状态, interactionObjectId 返回交互物体 Id 索引

6.1.2 手姿态识别状态

enum Hi5_Glove_Gesture_Recognition_State

{


```

ENone = 0,
EOK,
EFist,
    EIndexPoint,
EHandPlane
}

```

Hi5_Glove_Gesture_Recognition_State GetRecognitionState()

Hi5_Glove_Gesture_Recognition_State 返回手当前状态

6.1.3 手事件接口

```

public void MessageFun(string messageKey, object param1, object param2)
{
    if
(messageKey.CompareTo(Hi5_Glove_Interaction_Message.Hi5_MessageMessageKe
y.messageHandEvent) == 0)
    {
        Hi5_Glove_Interaction_Hand_Event_Data data = param1 as
Hi5_Glove_Interaction_Hand_Event_Data;

        switch (data.mEventType)
        {
        case EEventHandType.EClap:
        {
            //拍击事件
        }

            break;

        case EEventHandType.EPoke:
        {
            //戳事件
        }

            break;

        case EEventHandType.EPinch:
        {
            //抓取事件

```

```

    }

    break;
    case EEventHandType.EThrow:
    {
        //抛出事件
    }

    break;
    case EEventHandType.ELift:
    {
        //托举事件
    }

    break;
    case EEventHandType.ERelease:
    {
        //释放事件
    }

    break;
    }
}
}
}

```

6.2 交互物体接口

Hi5_Interface_Object

6.2.1 交互物体状态

enum E_Object_State

```

{
    ENone = -1,
    EStatic = 1,
    EPinch = 3,
    EMove = 2,
    EClap = 4,
    EFlyLift = 5,

```

EPoke = 6,

}

E_Object_State GetObjectItemState();获取交互物体状态

int GetObjectId(); 返回交互物体 Id

6.2.2 交互物体事件

```
public void MessageFun(string messageKey, object param1, object param2)
{
    if
(messageKey.CompareTo(Hi5_Glove_Interaction_Message.Hi5_MessageMessageKe
y.messageObjectEvent) == 0)
    {
        Hi5_Glove_Interaction_Object_Event_Data data = param1 as
Hi5_Glove_Interaction_Object_Event_Data;
        if (data.mObjectId == ObjectItem.idObject)
        {
            switch (data.mEventType)
            {
                case EEventObjectType.EClap:
                {
                }
                break;
case EEventObjectType.EPoke:
                break;
                case EEventObjectType.EPinch:
                break;
                case EEventObjectType.EMove:
                break;
                case EEventObjectType.ELift:
                break;
                case EEventObjectType.EStatic:
                if (mItem != null)
                {
```

```

                                mItem.ResetCorlor();
                                }
                                break;
                        }
                }
        }
}

```

6.2.3 按钮接口

Hi5_Interface_Button

```

virtual public void MessageFun(string messageKey, object param1, object param2)
{
    if
(messageKey.CompareTo(Hi5_Glove_Interaction_Message.Hi5_MessageMessageKe
y.messageObjectEvent) == 0)
    {
        Hi5_Glove_Interaction_Object_Event_Data data = param1 as
Hi5_Glove_Interaction_Object_Event_Data;
        if (data.mObjectId == ObjectItem.idObject)
        {
            if (data.mEventType == EEventObjectType.EClap)
            {

            }

        }
else if (data.mEventType == EEventObjectType.EPoke)
        {

        }
        else if (data.mEventType == EEventObjectType.EStatic)
        {

        }
    }
}

```

```
    }  
}
```

7.SDK API 调用

7.1Hi5 模块启动关闭接口

Hi5 模块启动和关闭，已默认调用，用户不必再重复调用。

Hi5_Thread_MonoBehaviour.cs 脚本提供。

- 启动 Hi5 手套功能：

```
void StartHi5()
```

- 关闭 Hi5 手套功能：

```
void StopHi5()
```

7.2Hi5 校准流程接口

HI5_Glove_Calibration_Process_Interface.cs 脚本提供 SDK 校准相关接口：

- 1) 校准姿态枚举：

```
public enum HI5_Pose  
{  
    Unknown,  
    VPose, //VPos 校准  
    BPose, //BPos 校准  
    PPose, //PPos 校准  
}
```

- 2) 手套校准指令接口：

- 进行对应姿态校准，具体参数参照上述校准姿态枚举值。

```
void StartCalibration(HI5_Pose pose)
```

- 获取校准进度接口：

- 3) 获取校准进度。

```
float GetCalibrationProgress(HI5_Pose pose)
```

7.3Hi5 相关骨节点和传感器数据

HI5_Glove_TransformData_Interface.cs 脚本提供 Hi5 骨节点和传感器数据相关接口。骨节点枚举：

```
public enum EHi5_Glove_TransformData_Bones  
{
```

```

    HandThumb1,
    HandThumb2,
    HandThumb3,
    InHandIndex,
    HandIndex1,
    HandIndex2,
    HandIndex3,
    InHandMiddle,
    HandMiddle1,
    HandMiddle2,
    HandMiddle3,
    InHandRing,
    HandRing1,
    HandRing2,
    HandRing3,
    InHandPinky,
    HandPinky1,
    HandPinky2,
    HandPinky3,
    NumOfHI5Bones,
}

```

7.4获取到手骨节点数据

1) 获取左手骨节点旋转和位置信息

```

Dictionary<EHi5_Glove_TransformData_Bones, Transform>
GetLeftHandTransform();

```

2) 获取右手骨节点旋转和位置信息

```

Dictionary<EHi5_Glove_TransformData_Bones, Transform>
GetRightHandTransform();

```

7.5获取传感器信息

1) 传感器信息定义

```

class HI5SensorInfor
{
    public int MagneticValue;    //磁信息
}

```

```

    public int EnergyValue;      //电信息
    public int SignalValue;      //信号信息
};

```

2) 获取左手传感器信息

```
Dictionary<EHi5_Glove_Sensor, HI5SensorInfor> GetLeftHandSensors();
```

3) 获取右手传感器信息

```
Dictionary<EHi5_Glove_Sensor, HI5SensorInfor> GetRightHandSensors();
```

4) 获取振子传感器信息

```
HI5SensorInfor GetRumblerSensorInfor(int RumblerIndex);
```

- RumblerIndex: 振子索引号, 值域[1,4], 分别对应振子 1 号--振子 4 号

7.6Hi5 振子振动接口

HI5_Glove_TransformData_Interface.cs 脚本函数 Vibrate 提供振子振动接口

1) 使振子振动:

```
void Vibrate(int rumblIndex, int vibrateLevel, uint duration, uint times);
```

2) 参数介绍:

- rumblIndex: 输入 1-4 代表振子索引号, 值域[1,4], 分别对应振子 1 号--振子 4 号;输入 0 所有振子全振动
- vibrateLevel: 振子振动等级, 1.弱振动 2.中级振动 3.强振动
- duration: 振动时间, 值域[100, 25500] (100 的整数倍), 单位毫秒
- Times: 振动次数, 值域[1,255]