Hi5-2 交互 SDK 用户手册

——Pico Neo3

目录

一、 Unity VR 基础环境配置

建议使用 Unity 2019.x/2020.x/2021.x LTS 版本新建工程，目前只支持 Pico Neo3 ，不支持 Pico Neo2，Unity 2022 版本正在适配中。

1. 插件下载

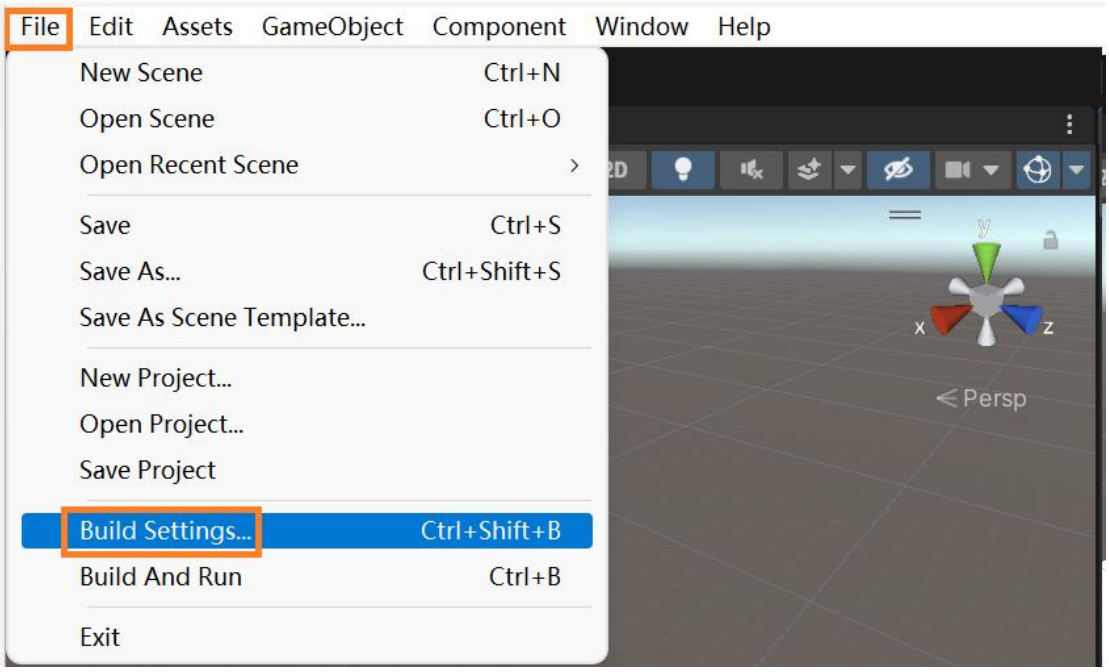
到 Pico 官网 https://developer.pico-interactive.com/sdk 下载最新版本 Unity XR SDK

pico vr unity 快速开发文档 https://developer.pico-interactive.com/document/doc

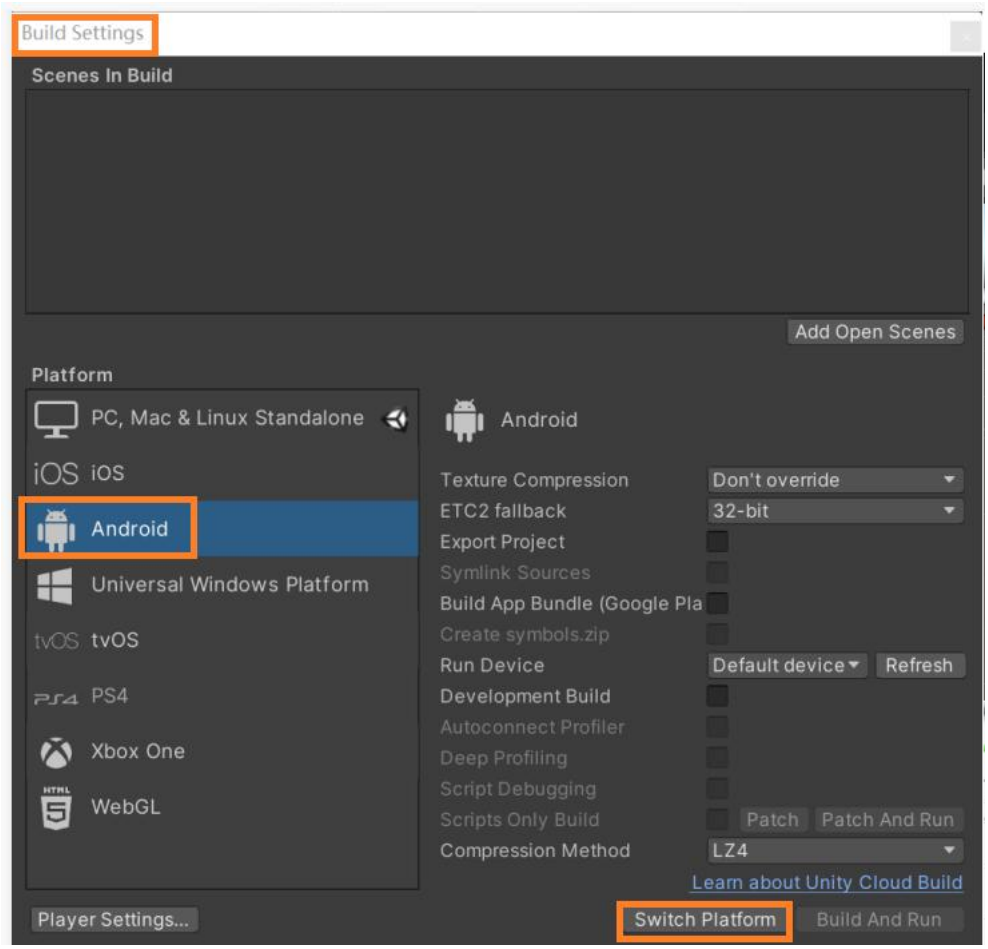
2. 插件安装

1) 解压 SDK 文件 Pico UnityXR SDK v2.0.5.zip

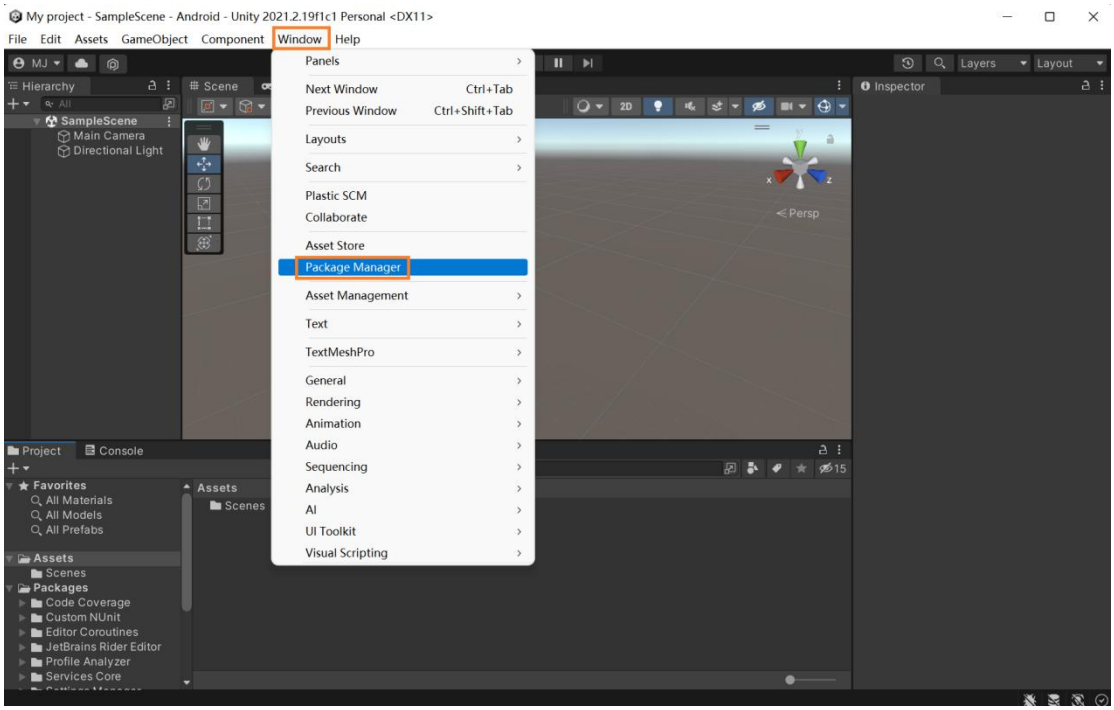
2) 新建或打开 Unity 工程，切换到 Android 平台，点击 File ->Build Settings



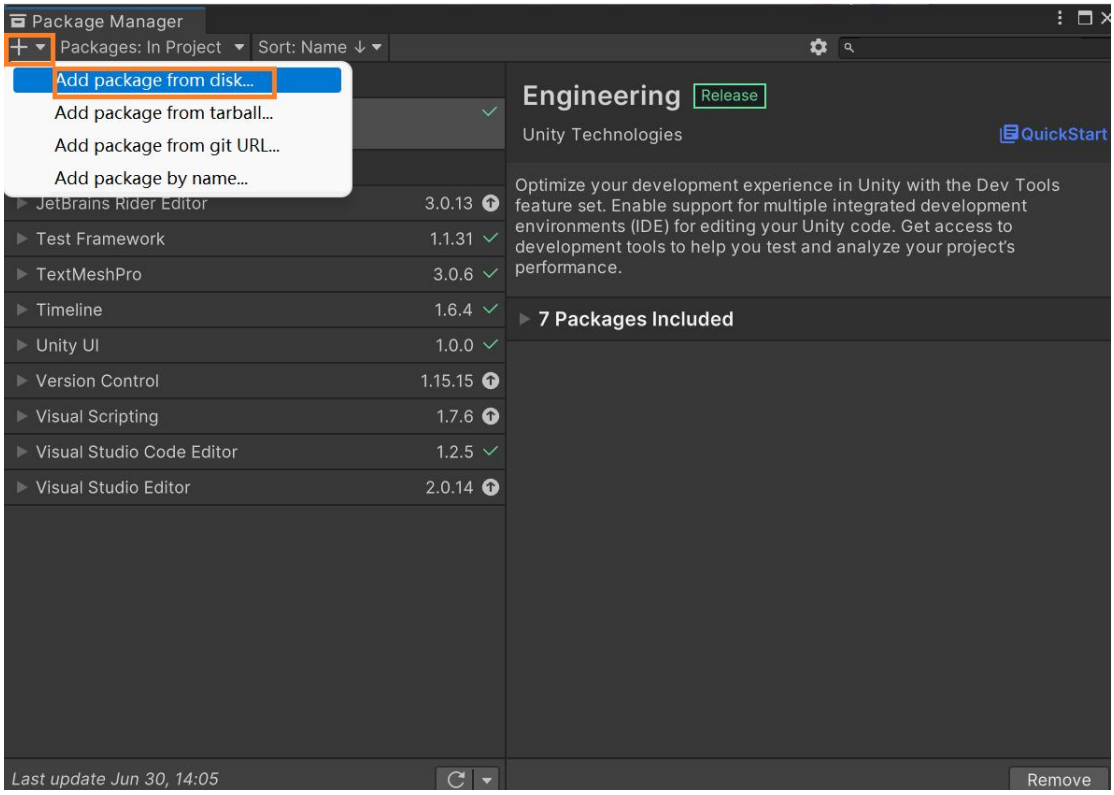
Build Settings 窗口中选择 Android 平台，然后点击 Switch Platform 按钮

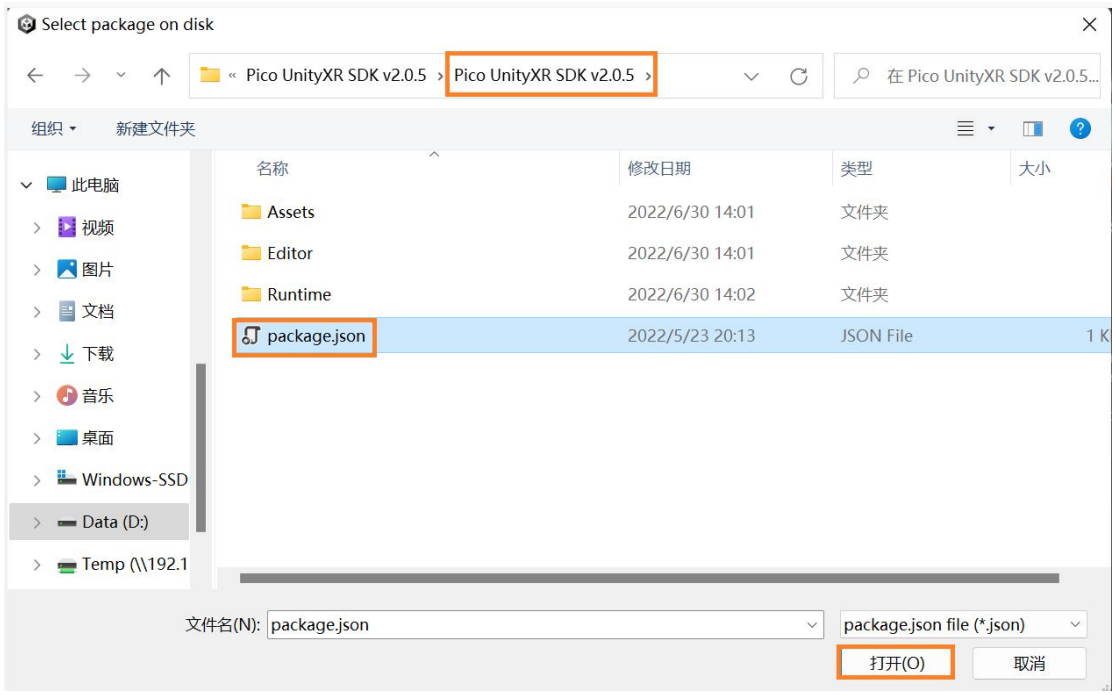


3) 点击 Windows ->Package Manager 打开 Package Manager 窗口，导入 Pico Unity XR SDK，如图所示：

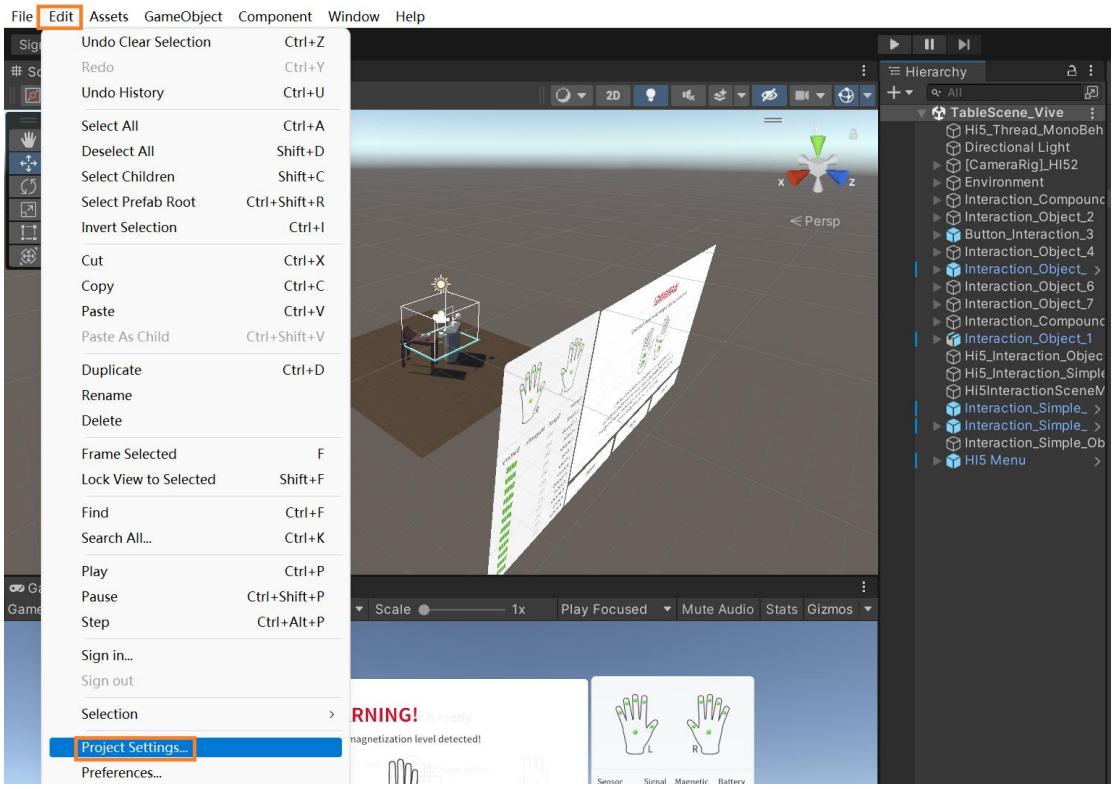


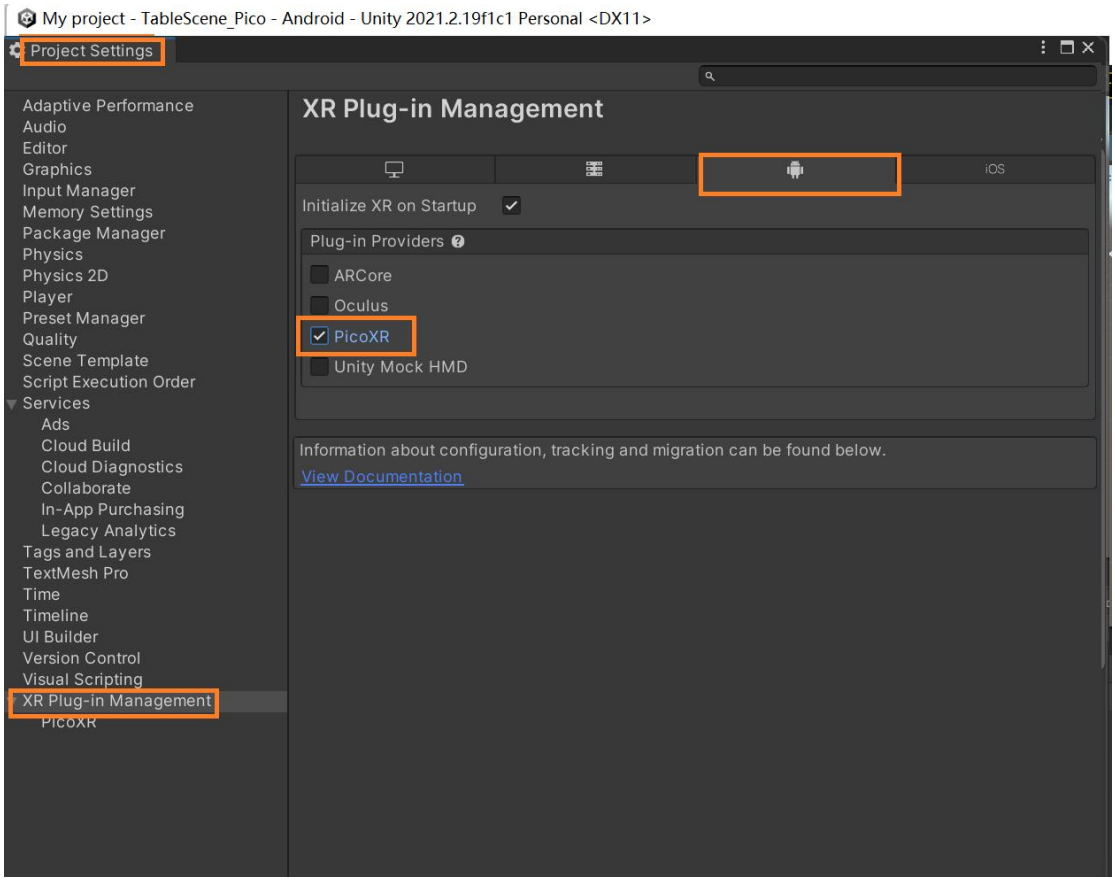
Package Manager ->Add package from disk… ,导入 package.json 文件,如图所示:





4) 导入成功后点击 Edit ->Project Settings，应用 PicoXR 插件，如图所示:


File
Edit
Assets
GameObject
Component
Window
Help
Undo Clear Selection Ctrl+Z
Redo Ctrl+Y
Undo History Ctrl+U
Select All Ctrl+A
Deselect All Shift+D
Select Children Shift+C
Select Prefab Root Ctrl+Shift+R
Invert Selection Ctrl+I
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Paste As Child Ctrl+Shift+V
Duplicate Ctrl+D
Rename
Delete
Frame Selected F
Lock View to Selected Shift+F
Find Ctrl+F
Search All... Ctrl+K
Play Ctrl+P
Pause Ctrl+Shift+P
Step Ctrl+Alt+P
Sign in...
Sign out
Selection
Project Settings...
Preferences...
Hierarchy
TableScene_Vive
Persp
Scale
1x
Play Focused
Mute Audio
Stats
Gizmos


My project - TableScene_Pico - Android - Unity 2021.2.19f1c1 Personal <DX11>
Project Settings
Adaptive Performance
Audio
Editor
Graphics
Input Manager
Memory Settings
Package Manager
Physics
Physics 2D
Player
Preset Manager
Quality
Scene Template
Script Execution Order
Services
Ads
Cloud Build
Cloud Diagnostics
Collaborate
In-App Purchasing
Legacy Analytics
Tags and Layers
TextMesh Pro
Time
Timeline
UI Builder
Version Control
Visual Scripting
XR Plug-in Management
PicoXR
XR Plug-in Management
iOS
Initialize XR on Startup
Plug-in Providers
ARCore
Oculus
PicoXR
Unity Mock HMD
Information about configuration, tracking and migration can be found below.
View Documentation

二、 Hi5_2 交互 SDK 安装

Unity VR 环境配置完成后，先安装 Hi5-2 SDK，再安装交互 SDK。

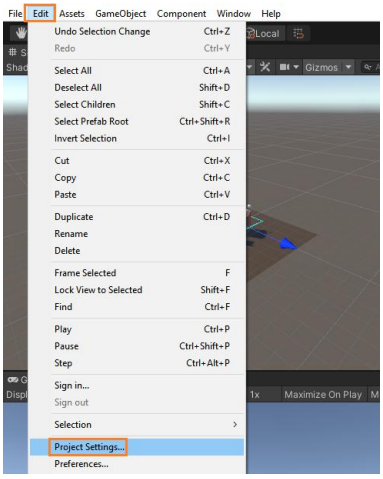
1) 先导入 Hi5-2 SDK ：**Hi5_2_Package_Pico.unitypackage**

2) 再导入交互 SDK：**Hi5_2_Interaction_Pico.unitypackage**

三、 Hi5_2 交互 SDK 应用

1. 工程设置

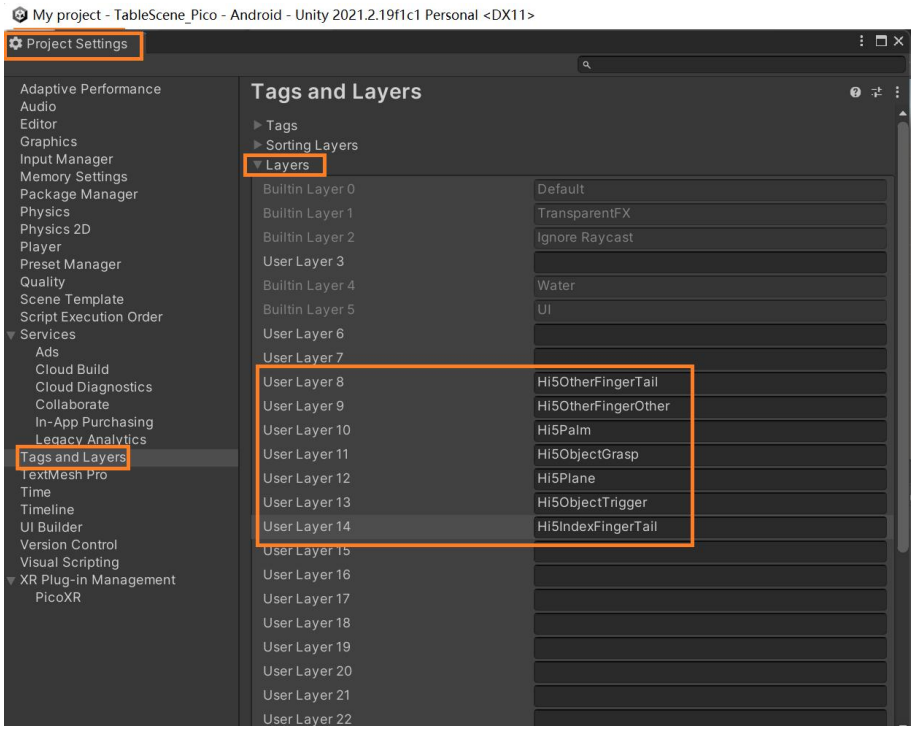
点击 Edit ->Project Settings 打开工程设置窗口，如图所示：



1.1 设置 Tags and Layers

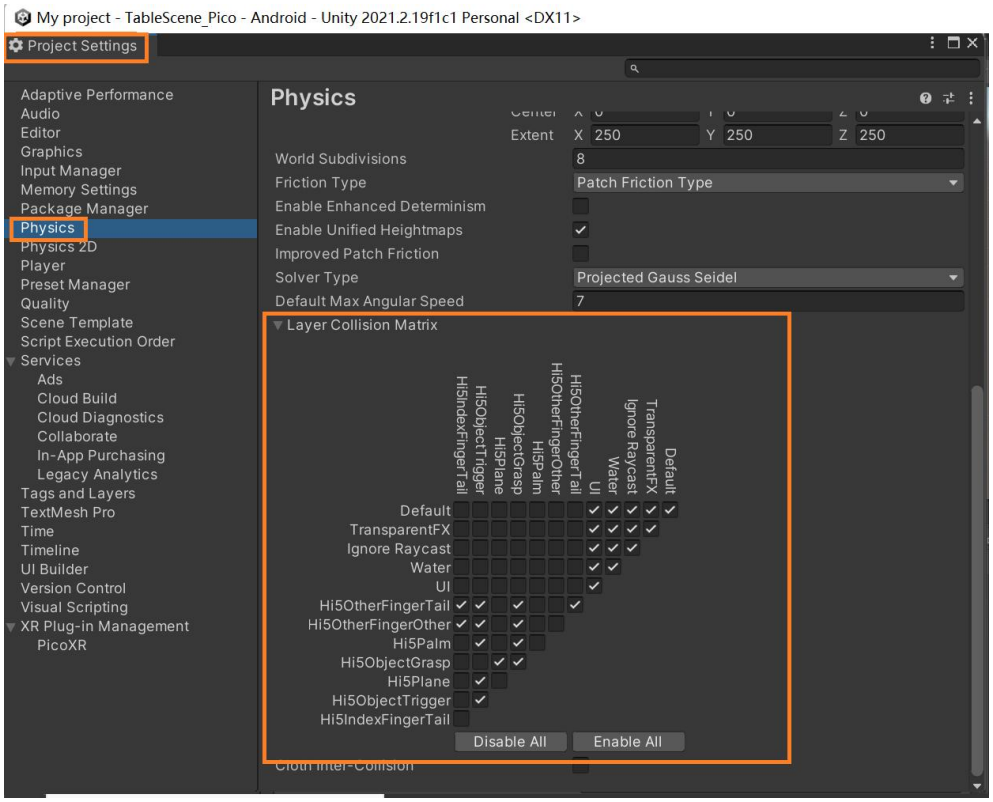
如图所示：

Layer 8	Hi5OtherFingerTail
Layer 9	Hi5OtherFingerOther
Layer 10	Hi5Palm
Layer 11	Hi5ObjectGrasp
Layer 12	Hi5Plane
Layer 13	Hi5ObjectTrigger
Layer 14	Hi5IndexFingerTail



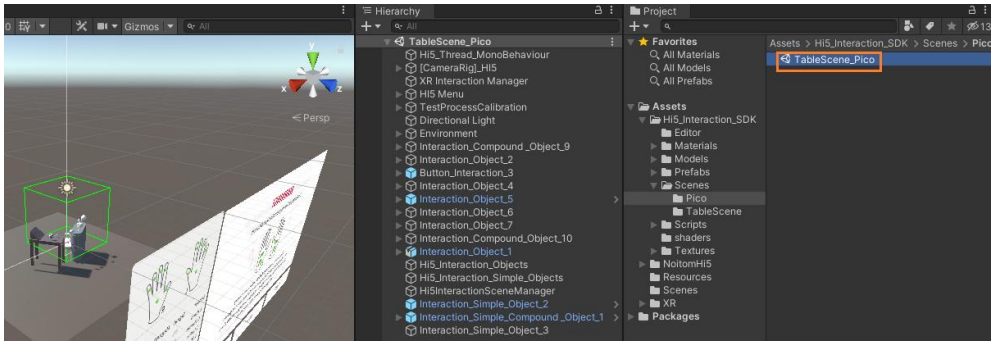
1.2 设置 Physics

如图所示：



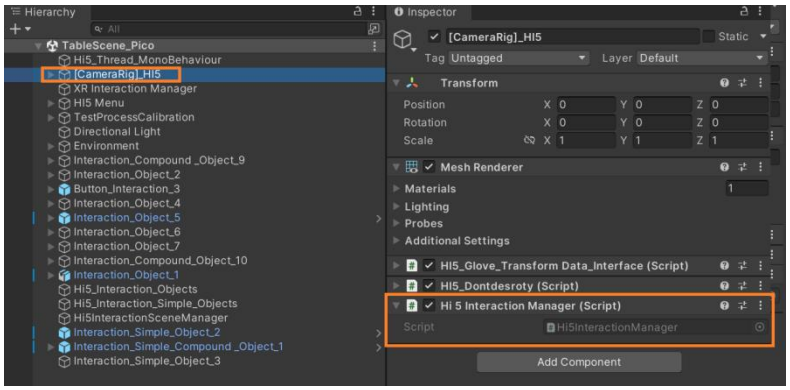
2. 场景设置

打开示例场景 TableScene_Pico，参考其设置，如图所示：

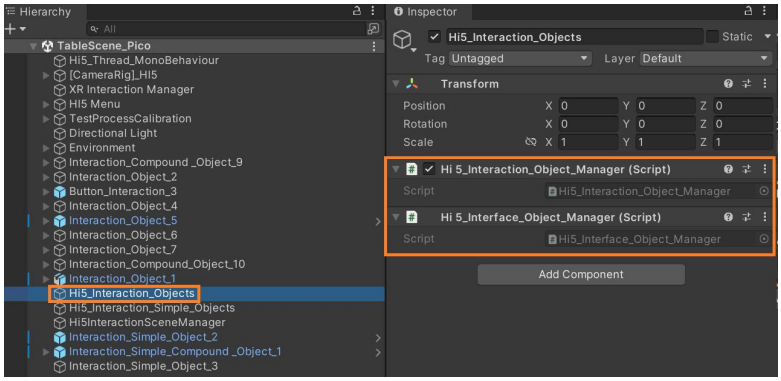


场景中必须包含以下内容：

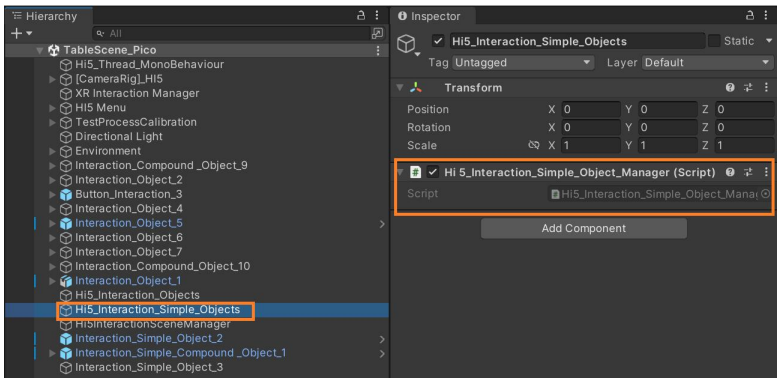
1) Hi5 Interaction Manager，如图所示：



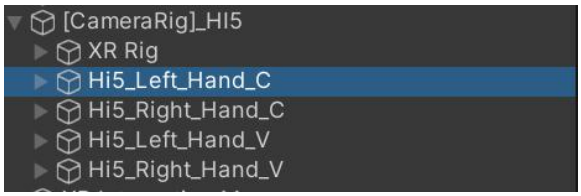
2) Hi5_Interaction_Objects，如图所示：



3) Hi5_Interaction_Simple_Objects，如图所示：



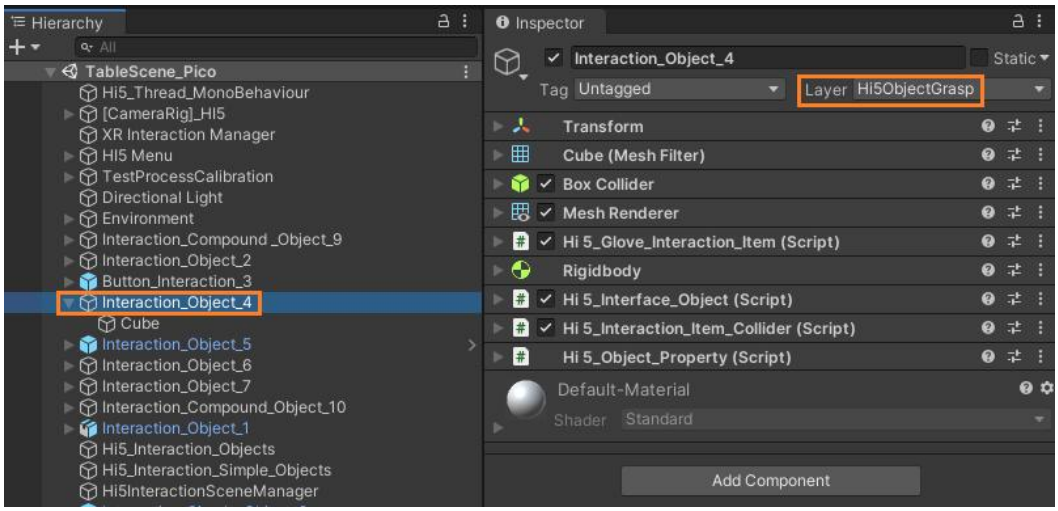
4) Hi5_Left_Hand_C ，Hi5_Right_Hand_C ，Hi5_Left_Hand_V ，Hi5_Right_Hand_V，如图所示：

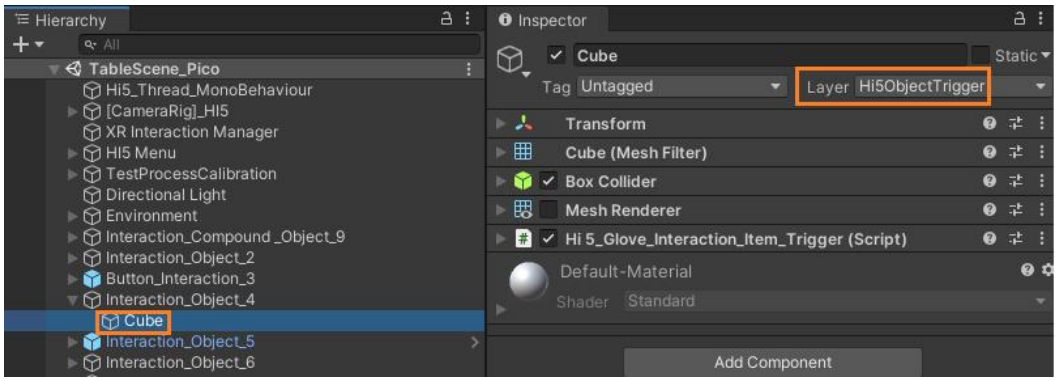


3. 物体设置

3.1 普通交互物体设置

物体设置分为父级物体和子级物体设置，父级物体 Layer 要设置为 Hi5ObjectGrasp，子级物体 Layer 要设置为 Hi5ObjectTrigger，例如 Interaction_Object_4 ->Cube，设置如图所示：

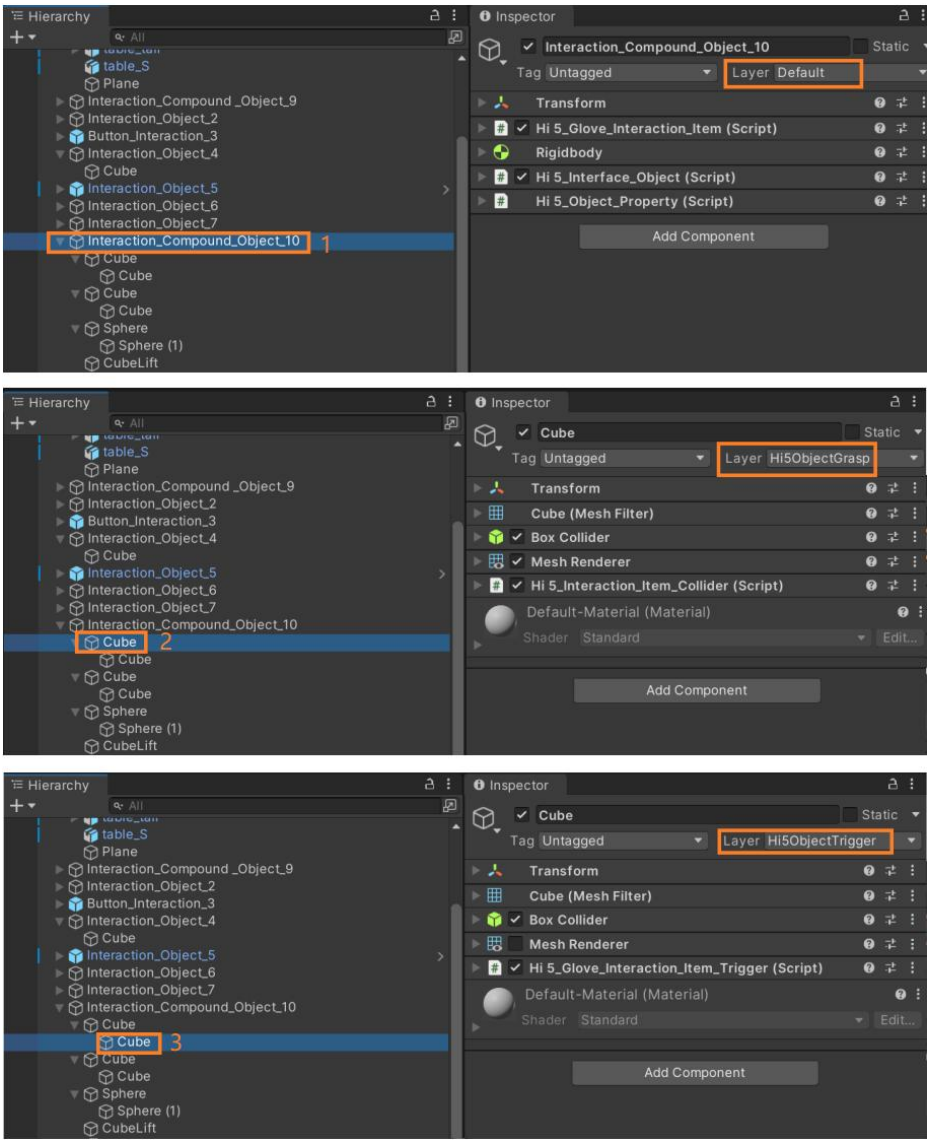




3.2 组合物体设置

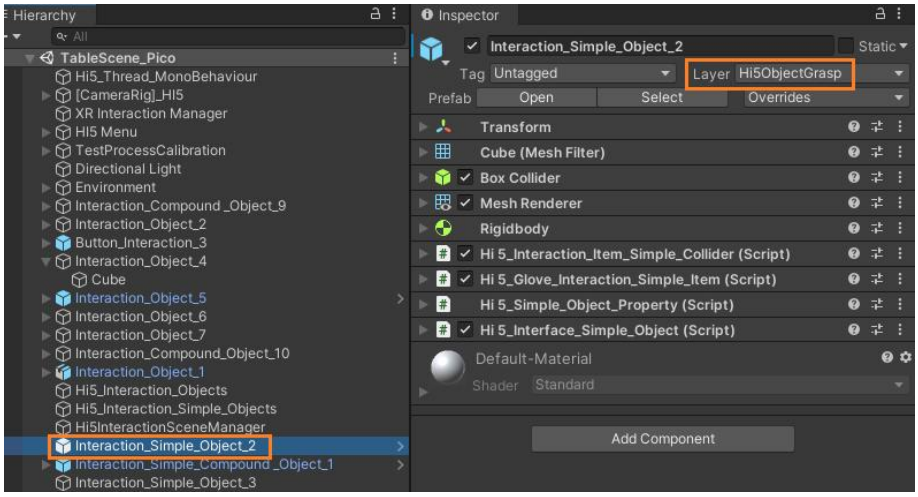
组合物体分为三层，组合物体的最外层物体 Layer 设置为 Default，组合物体中的外层物体的 Layer 设置为 Hi5ObjectGrasp，组合物体中的内层物体的 Layer 设置为 Hi5ObjectTrigger，

例如 Interaction_Compound_Object_10，设置如图所示：



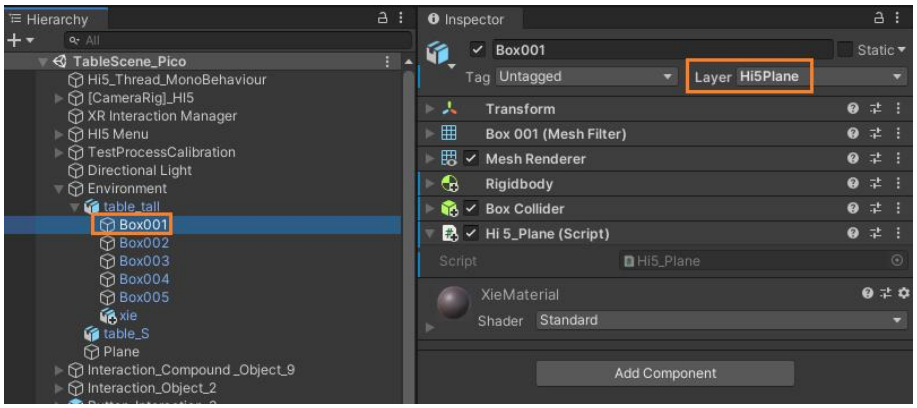
3.3 简单物体设置

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，其 Layer 要设置为 Hi5ObjectGrasp，例如 Interaction_Simple_Object_2，设置如图所示：



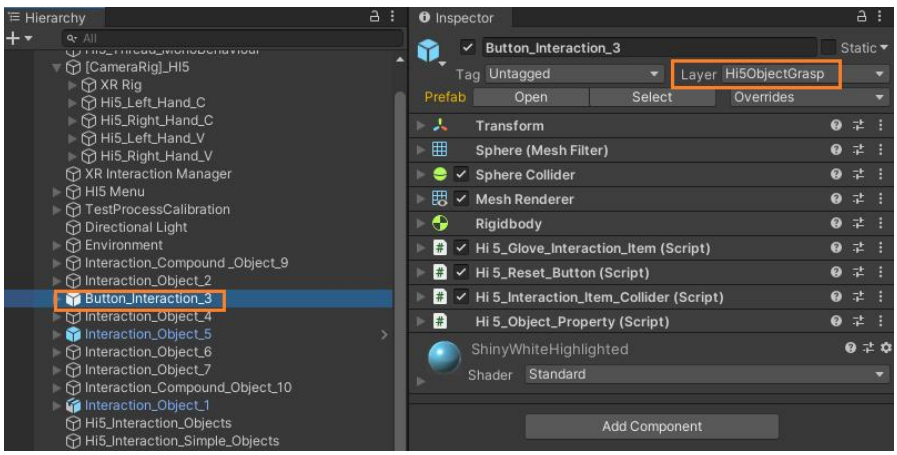
3.4 桌面物体

桌面放置物体的 Layer 要设置为 Hi5_Plane，例如 Box001，设置如图所示：



4. 按钮设置

例如 Button_Interaction_3，设置如图所示：



四、 相关接口

1. 手相关接口

Hi5_Interface_Hand 脚本

一、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

二、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

2. 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;
                }
            }
        }
```

3. 交互物体接口

Hi5_Interface_Object

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

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;
                }
            }
        }
    }
```

4. 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)
                    {
                    }
              }
          }
      }
```