

GSXR Unity SDK

开发者快速入门文档

V1.0.8

2021 年 11 月 03 日

目录

1. SDK 简介.....	1
2. SDK 配置说明.....	3
2.1. 支持设备.....	3
2.2. 开发环境要求.....	3
3. 开发注意事项.....	5
3.1. Android Manifest 文件配置.....	5
4. 快速入门.....	7
4.1. 准备文件.....	7
4.2. 新建工程.....	7
4.3. 打开包管理器.....	8
4.4. 包管理器切换筛选本地视图.....	8
4.5. 包管理器导入本地插件安钮.....	9
4.6. 选择文件导入.....	9
4.7. 导入完成.....	10
4.8. 根据实际需求导入 TextMesh 演示 Samples 所需资源.....	10
4.9. 快速转换平台.....	11
4.10. 一键生成默认设置.....	12
4.11. 普通相机一键转成 GSXR.....	13
4.12. 编写输入行为编辑器打开方式.....	14
4.13. 配置输入行为映射.....	14
4.14. 编辑手柄预制.....	15
4.15. 行为映射绑定.....	15
4.16. 生成 GSXR_Settings.....	16
4.17. 修改 GSXR_Settings.....	16
4.18. 调式运行.....	17
4.19. 自定义添加 DontDestroyOnLoad.....	18
4.20. 打包设置.....	18
5. 编辑行为.....	19
5.1. 行为编辑器.....	19
5.2. 行为绑定.....	20
5.2.1. 组件添加行为绑定集合.....	20
5.2.2. 组件添加行为绑定类型.....	20
5.2.3. Boolean 行为绑定路径和输入源.....	21
5.2.4. Single 行为绑定路径和输入源.....	22
5.2.5. Vector2 行为绑定路径和输入源.....	22
5.2.6. 自定义脚本绑定行为路径和输入源.....	23
5.2.7. 获取 Hmd 键值.....	24
5.2.8. 设置 GSXR_Manager 脚本执行优先级.....	24
6. 全局设置.....	25
6.1. 设置文件路径.....	25
6.2. 设置快捷指引.....	26
6.3. 设置参数说明.....	26
7. 异常处理.....	27

7.1. 版本异常.....	27
7.2. URP/LWRP 模板导入后 dll 拷贝失败.....	27
8. 注意事项.....	28
8.1. 设计返回逻辑.....	28

1. SDK 简介

本文档介绍 GSXR_SDK 在 Unity 引擎的实现方式，使用 GSXR_Unity_SDK(以下简称 SDK)制作运行在适配了 General Standard for XR (以下简称 GSXR)标准的 XR 设备上的 XR 应用。GSXR_Unity_SDK 主要包含：头部组件、手部组件和扩展多追踪器组件。每个组件都会包含定位模块，以及不同的交互方式。特殊之处是，头部组件还会包含图像模块，手部组件可以动态加载不同终端设备的模型。

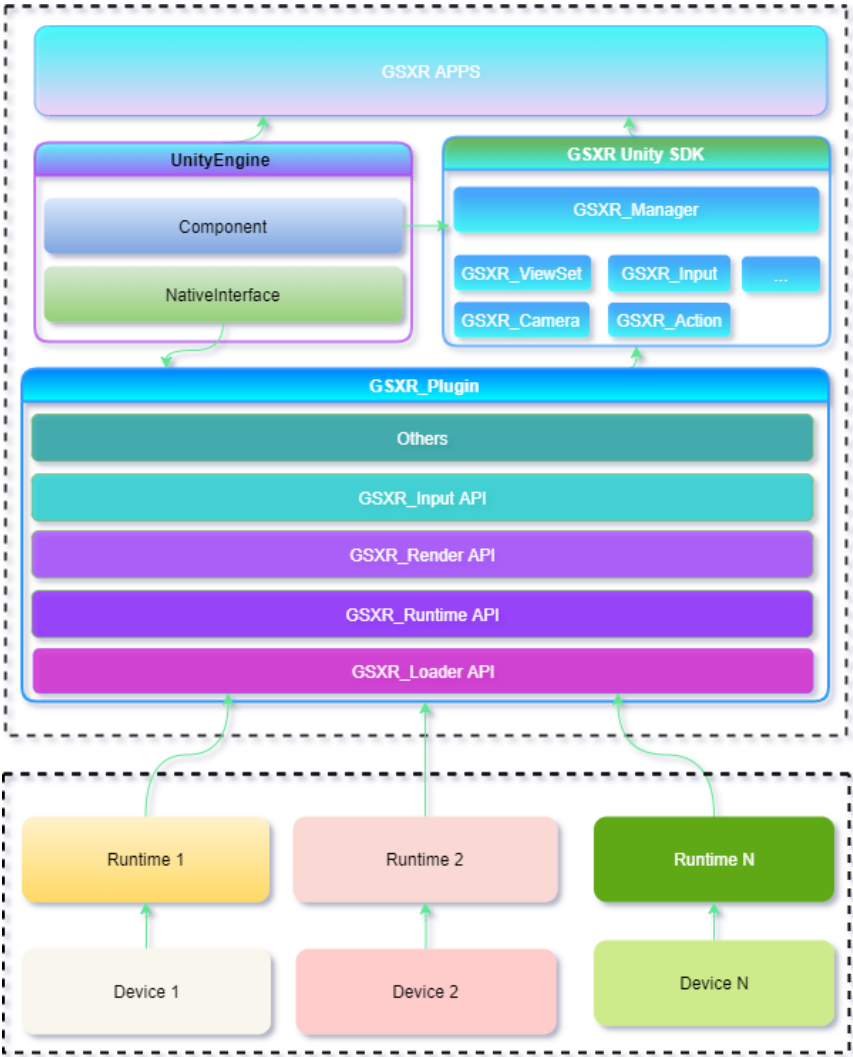


图 1 SDK 架构图

SDK 通过 UnityPlugin 文件的格式提供，开发者导入后可看到如下目录：

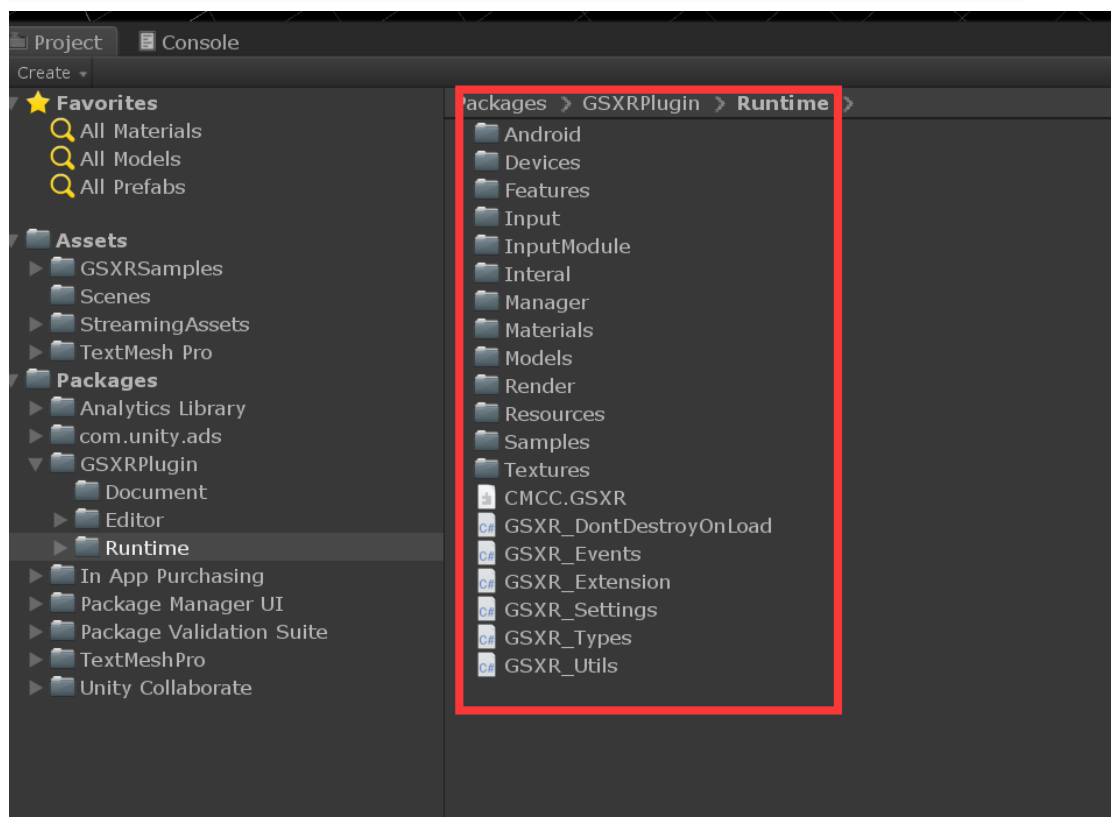


图 2 GSXRPlugin 目录结构

Packages->GSXRPlugin 下的每个子目录都对应 SDK 中相应的功能，子目录下的 Runtime/Samples 目录中提供了供您参考的场景和 GSXRSamples.unitypackage，在适配 SDK 时遇到问题可查看文档。

- GSXR_Settings 包含全局设置的参数相关功能；
- GSXR_Plugin 包含底层相关的接口相关功能；
- GSXR_Manager 包含 Runtime 与 Render 初始化的相关功能；
- GSXR_ViewSet 包含视图集渲染提交相关功能；
- GSXR_Camera 包含视图渲染提交相关功能；
- GSXR_Input 包含按键行为事件绑定相关功能；
- GSXR_TrackedObject 包含 Tracker 对象姿态追踪震动相关功能；
- GSXRInputModule 包含事件系统相关功能；
- GSXR_GazeSwitcher 包含射线切换相关功能。

2. SDK 配置说明

2.1. 支持设备

GSXR 系列设备

2.2. 开发环境要求

支持 Unity 版本：2018.4.x、2019x、2020x、2021x.

Supported Platforms and Low-Level Graphics APIs

Platform	D3D11	D3D12	OpenGL/GLES	Vulkan	Metal	Build Status
✓ ¹ Android	-	-	✓ ²			✓ ¹

##QualitySettings vSyncCount

Don't Sync	Every V Blank	Every Second V Blank
✓ ¹	-	-

##Minimum API level Recommended Setting

AndroidSDK
✓ ¹ >=AndroidApiLevel22

##Target Architectures

AndroidArchitecture	
✓ ¹ ARMv7	✓ ¹ ARM64

##Unity 2018 Template

Unity Version	2D	3D	3D Width Extras	Hight-Definition RP	Lightweight RP	VR Lightweight RP
Unity2018x	-	✓ ¹	✓ ¹	-	✓ ¹	-

##Unity 2019 Template

Unity Version	2D	3D	3D Width Extras	Hight-Definition RP	Universal RenderPipeline
Unity2019x	-	✓ ¹	✓ ¹	-	✓ ¹

##Unity 2020 Template

Unity Version	2D	3D	3D Width Extras	Hight-Definition RP	Universal RenderPipeline
Unity2020x	-	✓ ¹	✓ ¹	-	
✓ ¹					

##Unity 2021 Template

Unity Version	2D	3D	3D Width Extras	Hight-Definition RP	Universal RenderPipeline
Unity2021x	-	✓ ¹	✓ ¹	-	✓ ¹

图 3 版本支持

向下兼容存在项目风险请按支持版本进行移植。

JDK	jdk1.8.0 及以上
Android SDK	API Level 22 及以上
Android NDK	android-ndk-r16b (Unity2018.4x)
TextMeshPro (Plugin)	>=com.unity.textmeshpro@1.4.1
Scripts Runtime Version*	.NET 4.x Equivalent
Visual Studio	>=2017 (安装使用 Unity 开发游戏插件)

图 4 环境需求

注意：Unity 菜单 Windows->GSXR BuildSettings 推荐设置

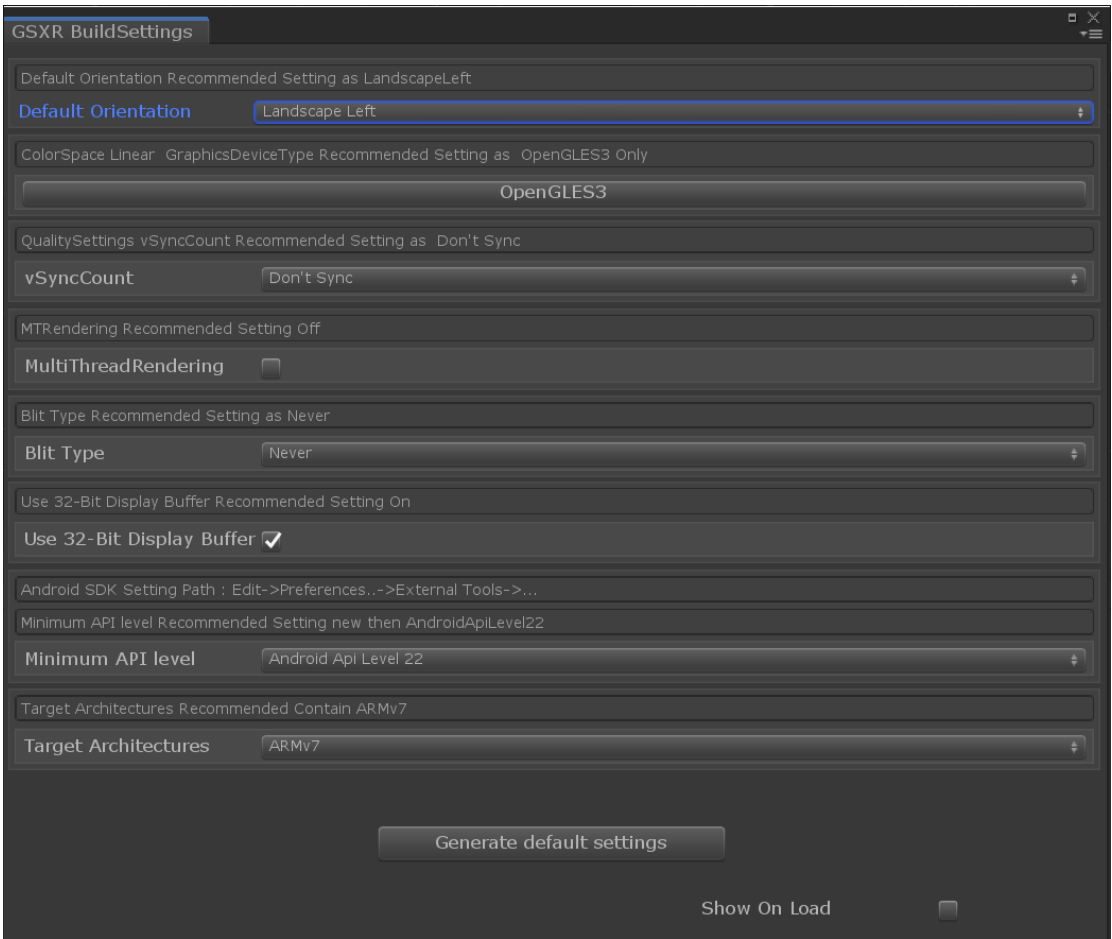


图 5 GSXR BuildSettings 界面

3. 开发注意事项

3.1. Android Manifest 文件配置

- 如果是 VR 应用需要添加特殊标签，实现 VR 模式显示，这个配置由不同厂商指定各自需求；

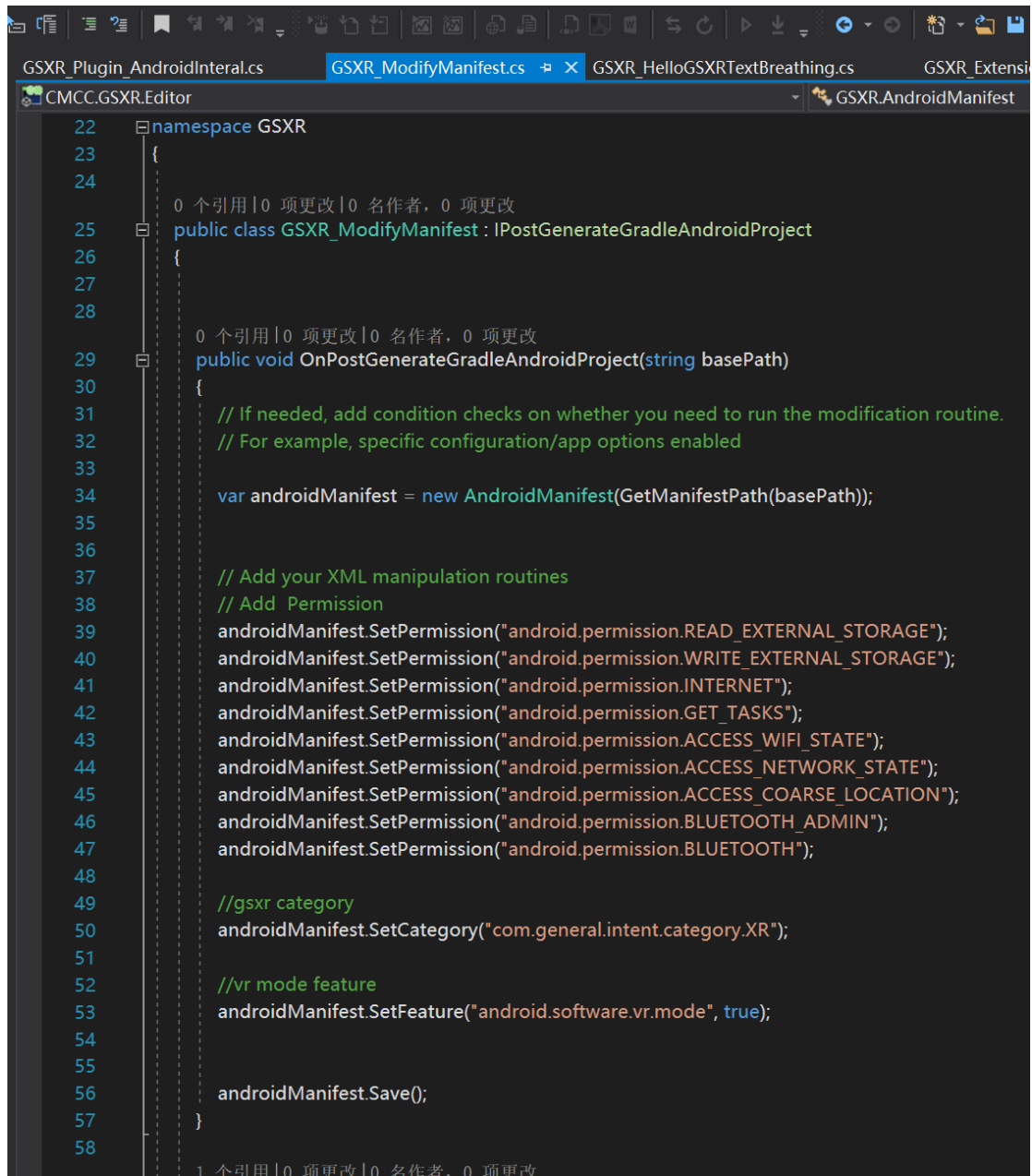
```
<uses-feature android:name="android.software.vr.mode" android:required="true" />
<category android:name="com.general.intent.category.XR" />
```

- 添加自定义的权限

```
<!--以下是必需权限-->
<!--SDK 基础功能-->
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<!--以下是可选权限-->
<!--蓝牙-->
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
<uses-permission android:name="android.permission.INJECT_EVENTS" />
<!--支付功能-->
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<!--震动接口-->
<uses-permission android:name="android.permission.VIBRATE"/>
<!--设置亮度 -->
<uses-permission android:name="android.permission.WRITE_SETTINGS" />
<!--修改语言 -->
<uses-permission android:name="android.permission.CHANGE_CONFIGURATION" />
```

- 修改方式

修改 GSXRplugin\Editor\GSXR_ModifyManifest.cs



```
22 namespace GSXR
23 {
24     0 个引用 | 0 项更改 | 0 名作者, 0 项更改
25     public class GSXR_ModifyManifest : IPostGenerateGradleAndroidProject
26     {
27         0 个引用 | 0 项更改 | 0 名作者, 0 项更改
28         public void OnPostGenerateGradleAndroidProject(string basePath)
29         {
30             // If needed, add condition checks on whether you need to run the modification routine.
31             // For example, specific configuration/app options enabled
32
33             var androidManifest = new AndroidManifest(GetManifestPath(basePath));
34
35             // Add your XML manipulation routines
36             // Add Permission
37             androidManifest.SetPermission("android.permission.READ_EXTERNAL_STORAGE");
38             androidManifest.SetPermission("android.permission.WRITE_EXTERNAL_STORAGE");
39             androidManifest.SetPermission("android.permission.INTERNET");
40             androidManifest.SetPermission("android.permission.GET_TASKS");
41             androidManifest.SetPermission("android.permission.ACCESS_WIFI_STATE");
42             androidManifest.SetPermission("android.permission.ACCESS_NETWORK_STATE");
43             androidManifest.SetPermission("android.permission.ACCESS_COARSE_LOCATION");
44             androidManifest.SetPermission("android.permission.BLUETOOTH_ADMIN");
45             androidManifest.SetPermission("android.permission.BLUETOOTH");
46
47             //gsxr category
48             androidManifest.SetCategory("com.general.intent.category.XR");
49
50             //vr mode feature
51             androidManifest.SetFeature("android.software.vr.mode", true);
52
53             androidManifest.Save();
54         }
55     }
56     1 个引用 | 0 项更改 | 0 名作者, 0 项更改
57
58 }
```

图 5 修改 Manifest 样例

4. 快速入门

4.1. 准备文件

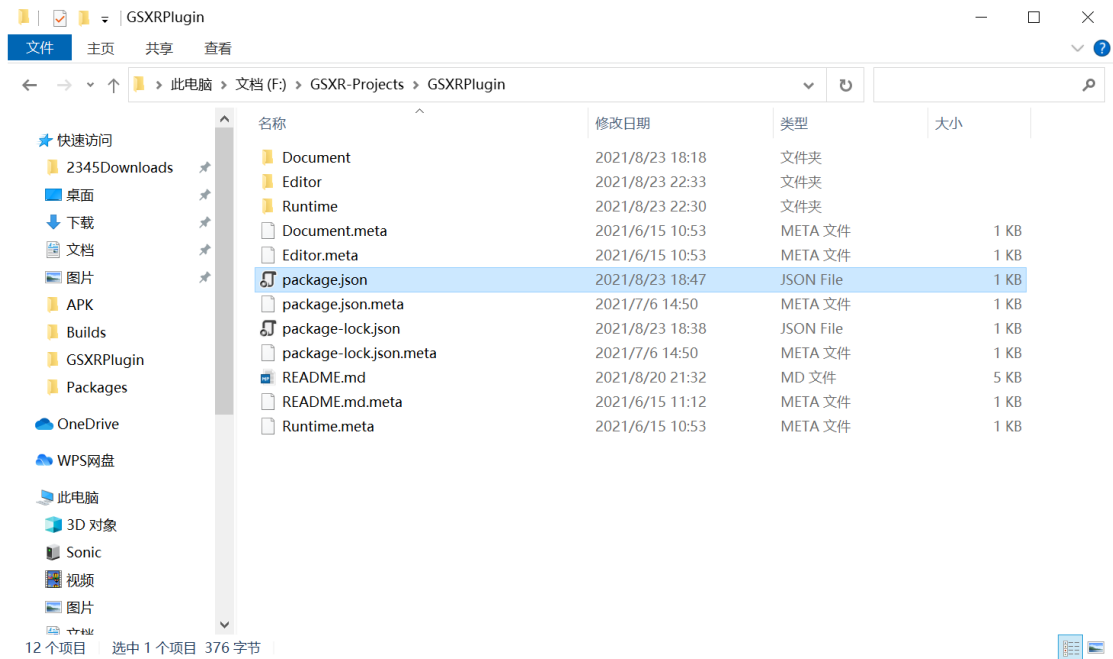


图 7 文件结构图

4.2. 新建工程

案例使用 Unity2018.4.3f1 3D 模板新建工程，如果有特殊需要可选择其他模板。

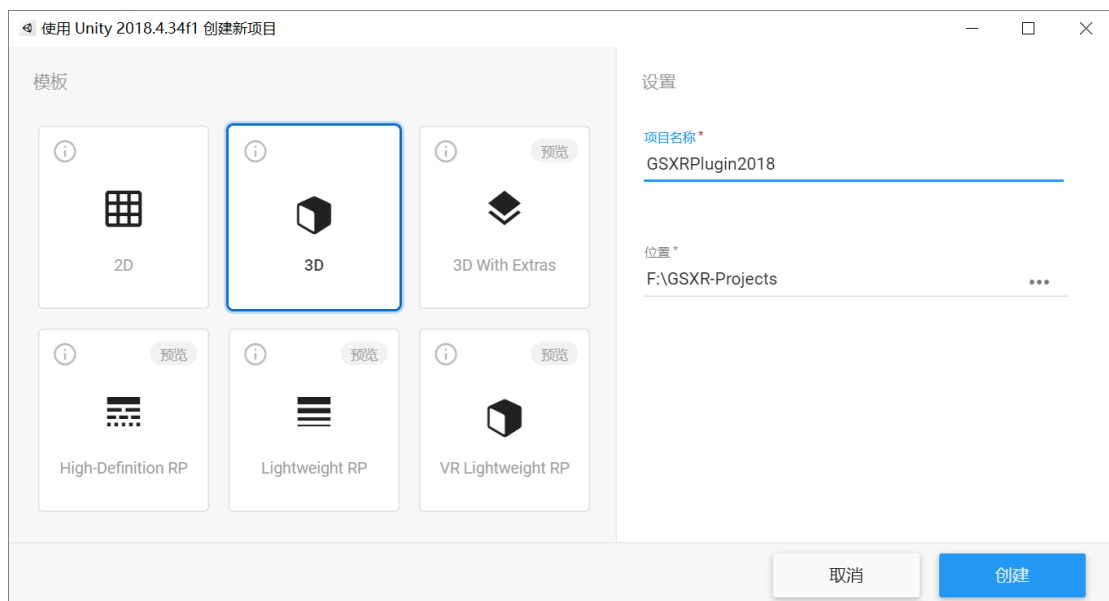


图 8 新建工程

4.3. 打开包管理器

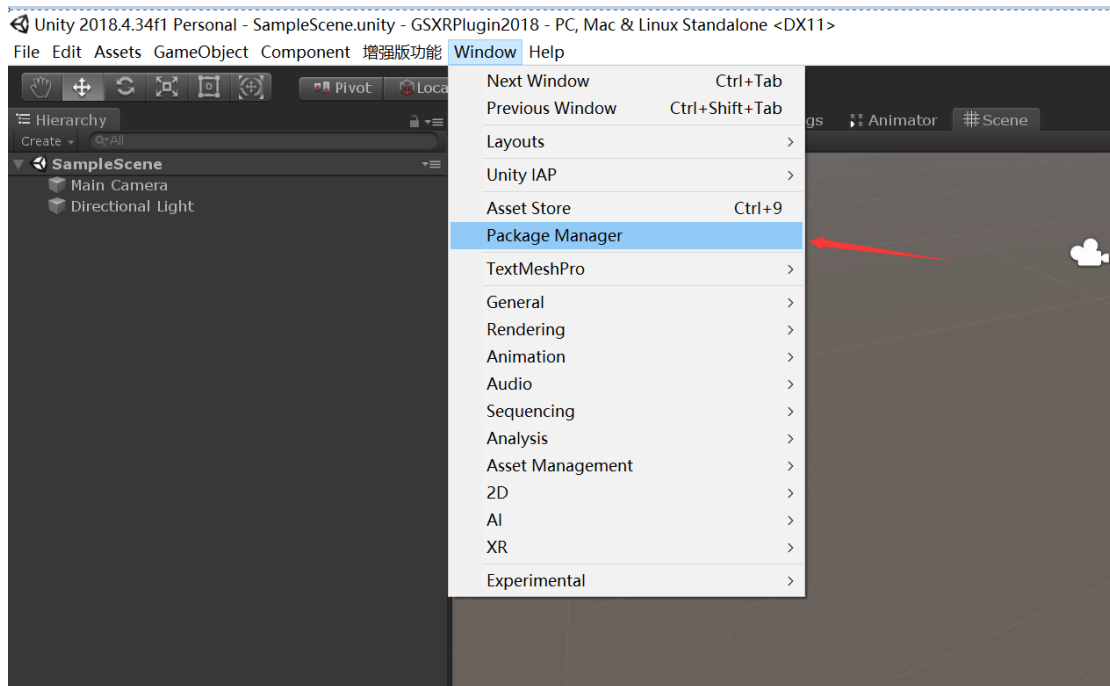


图 9 打开包管理器

4.4. 包管理器切换筛选本地视图

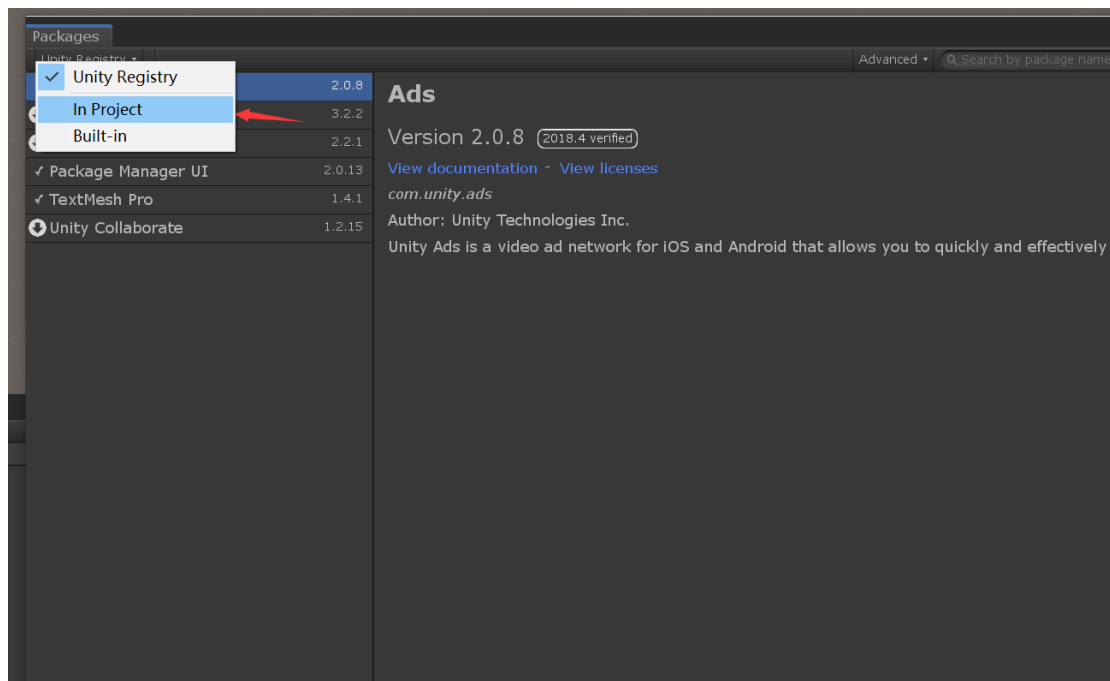


图 10 包管理器切换筛选本地视图

4.5. 包管理器导入本地插件按钮

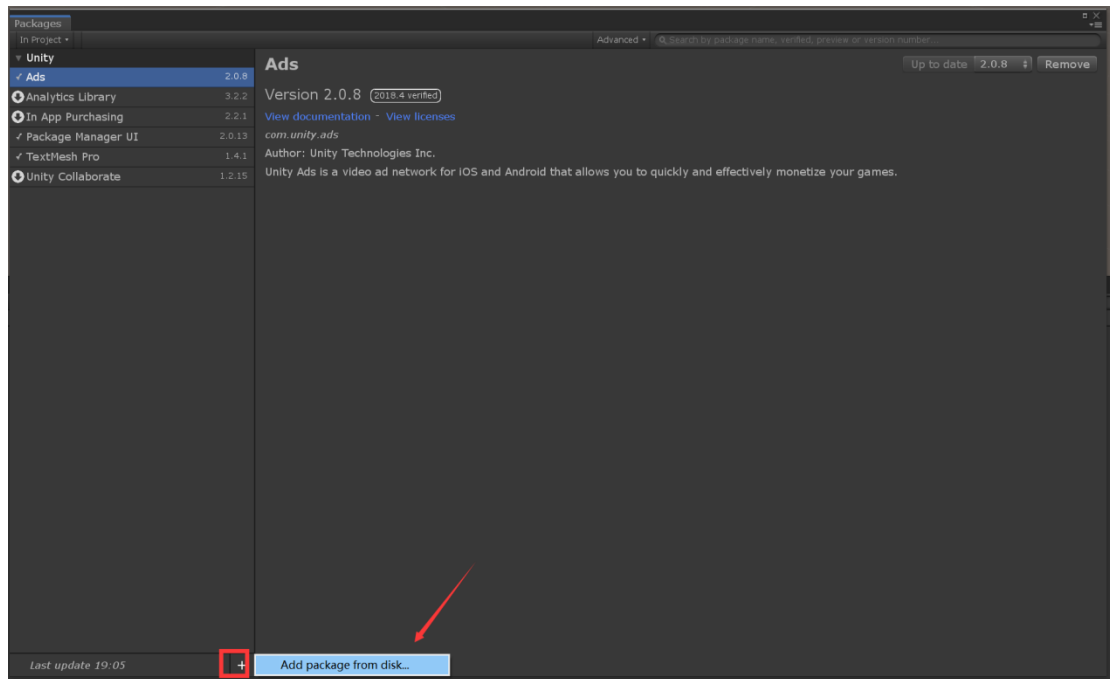


图 11 包管理器导入本地插件按钮

4.6. 选择文件导入

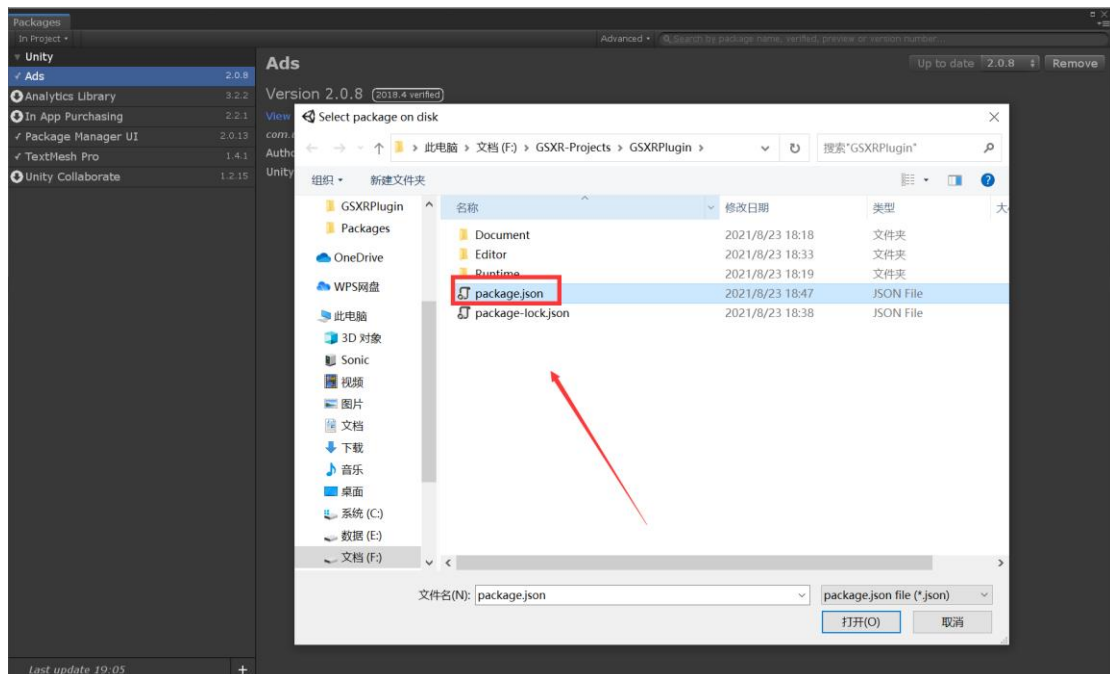


图 12 包管理器选择本地插件

4.7. 导入完成

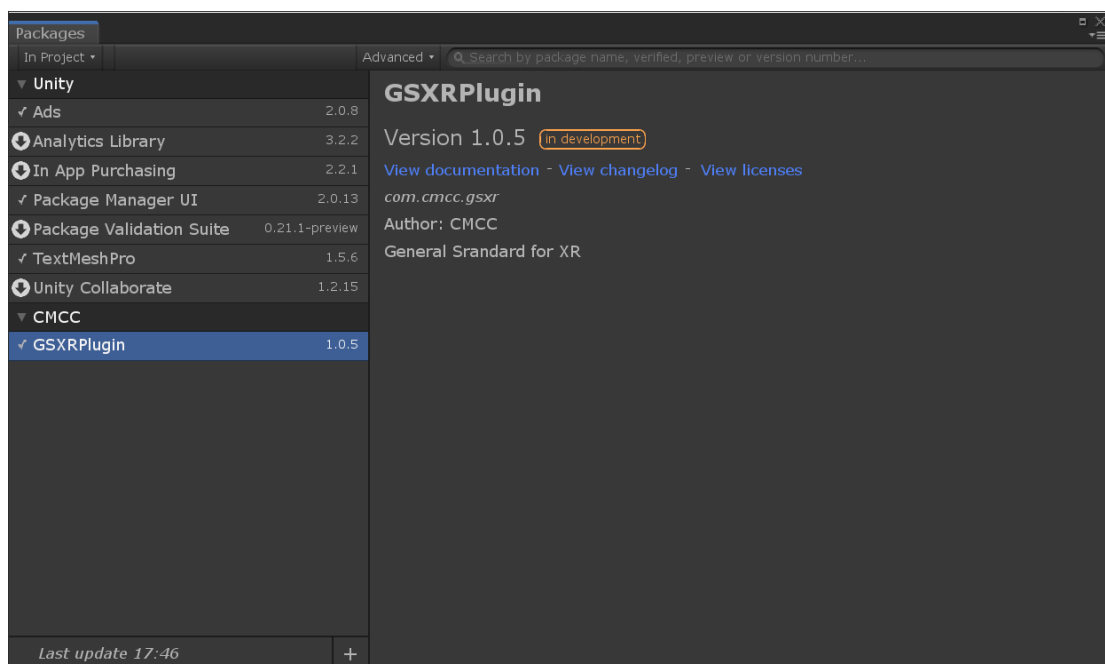


图 13 导入完成 GSXRPlugin 界面信息

4.8. 根据实际需求导入 TextMesh 演示 Samples 所需资源

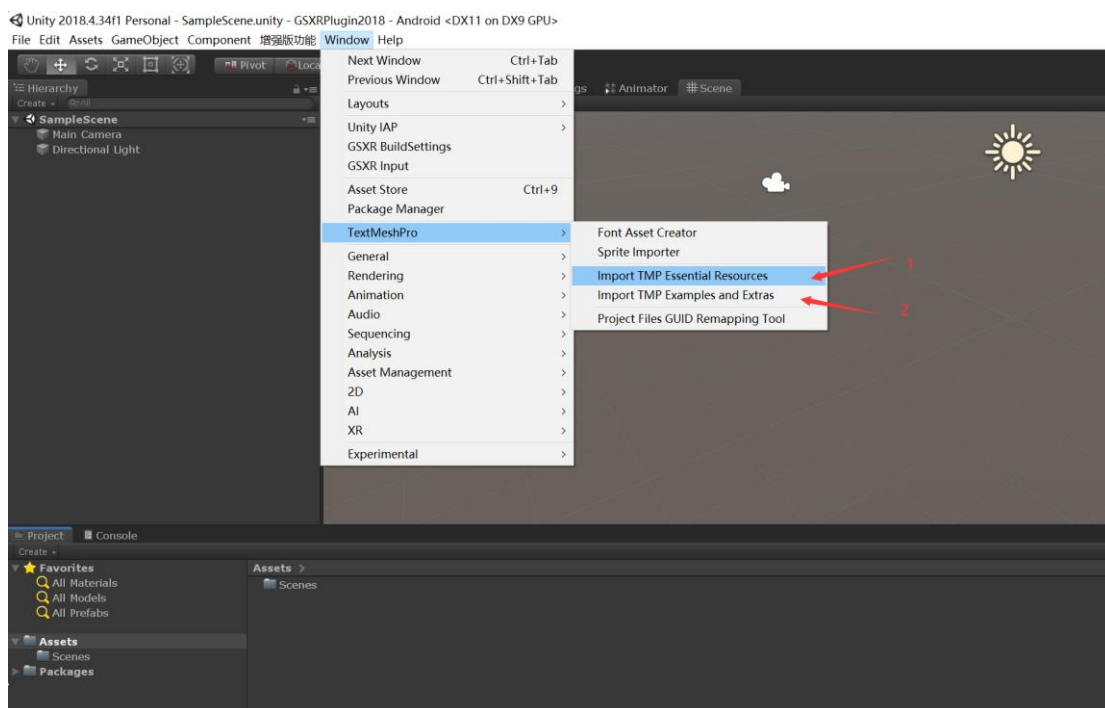


图 14 TextMeshPro 资源导入方法

4.9. 快速转换平台

GSXR 打包工具集成了平台转换和推荐参数自动切换的功能，当插件首次导入后如需快速配置，可通过打包工具一键设置。

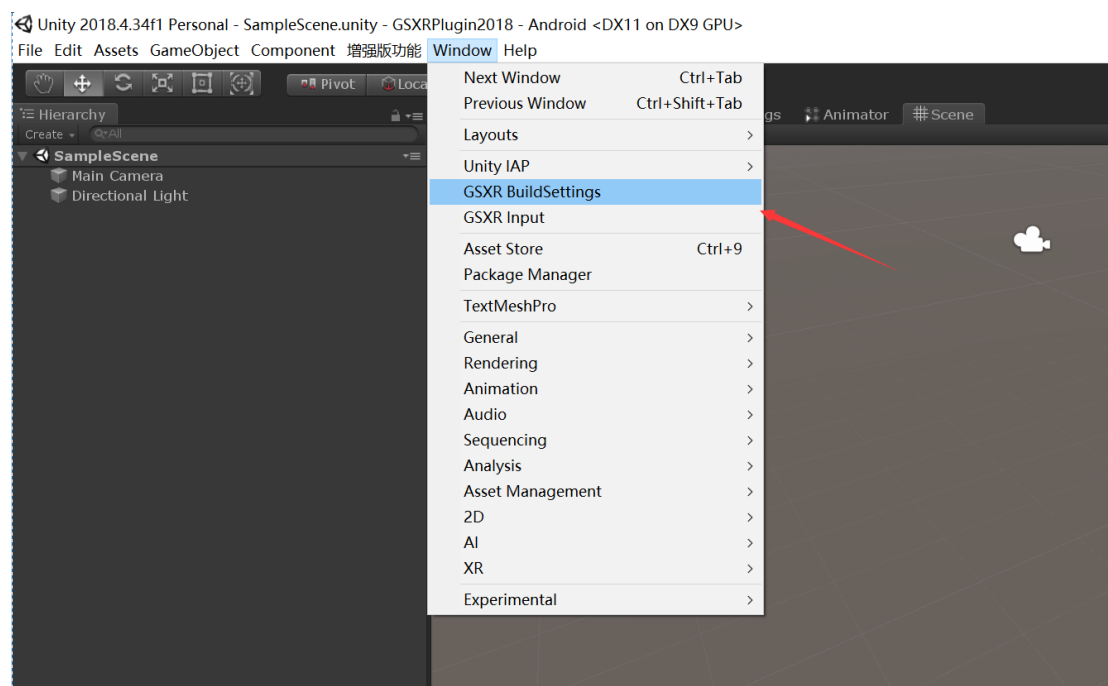


图 15 打开打包设置方式

4. 10. 一键生成默认设置

自动转换平台，设置推荐参数

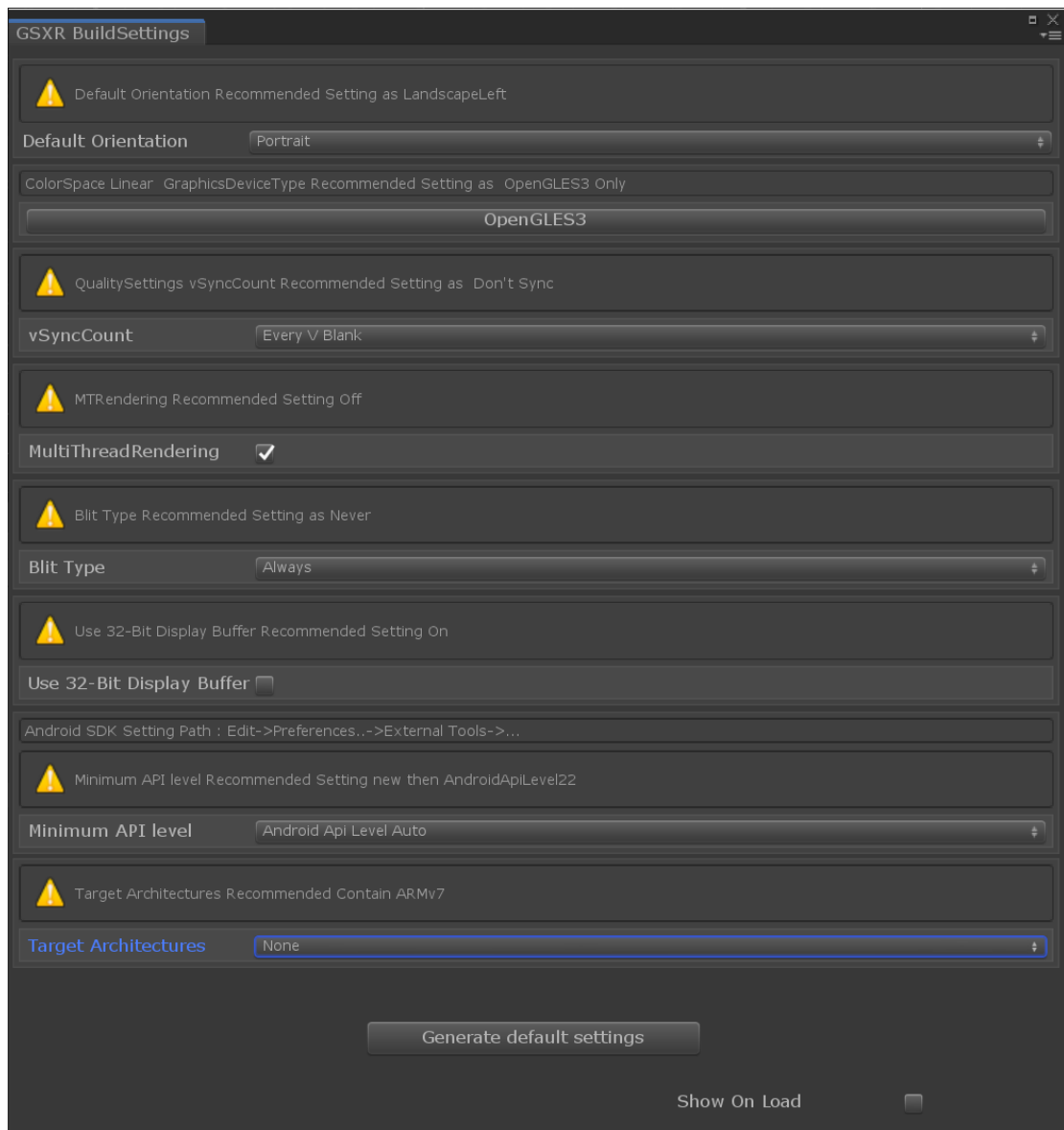


图 16 打包设置界面与推荐信息

4.11. 普通相机一键转成 GSXR

新建场景创建 GSXR 管理器，同步生成了必要组件和预制对象，开发者可通过编辑和修改组件参数达到自定义开发的目的。

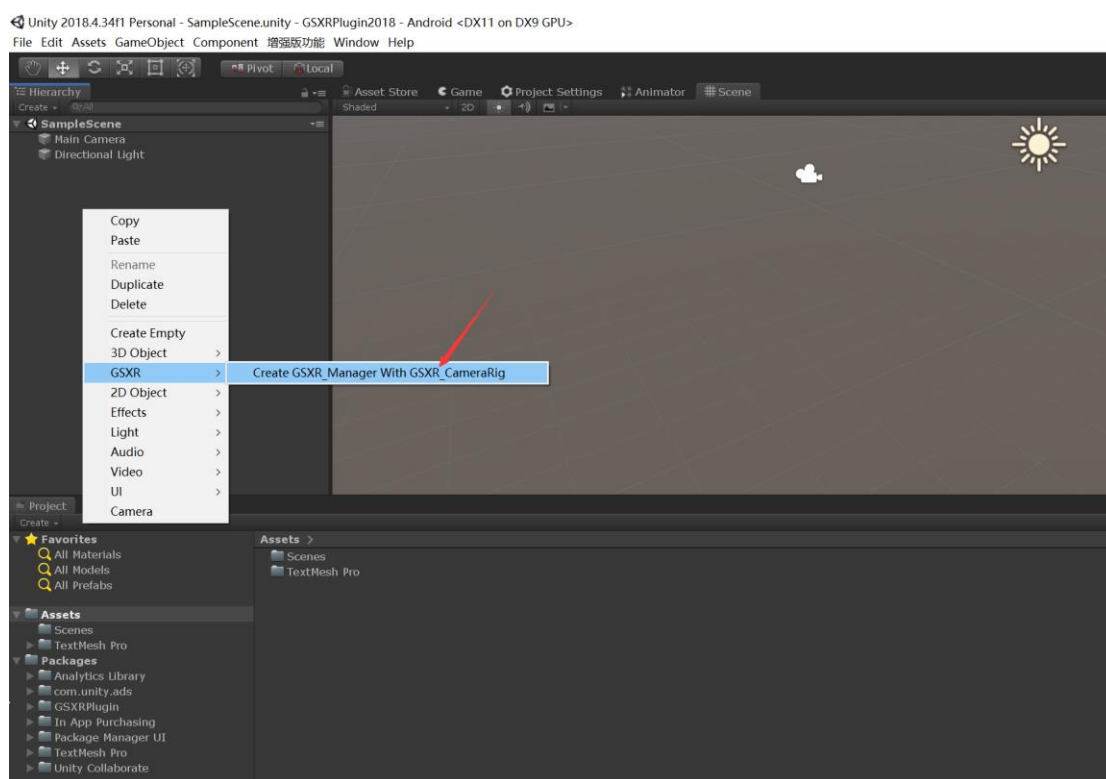


图 17 一键转成场景相机

GSXRManager 生成后有一个 MainCamera 开发者可根据需求调节它的参数或者绑定相关后期脚本，在程序运行后管理器会自动加载克隆当地相机的组件，进入相关模式渲染。

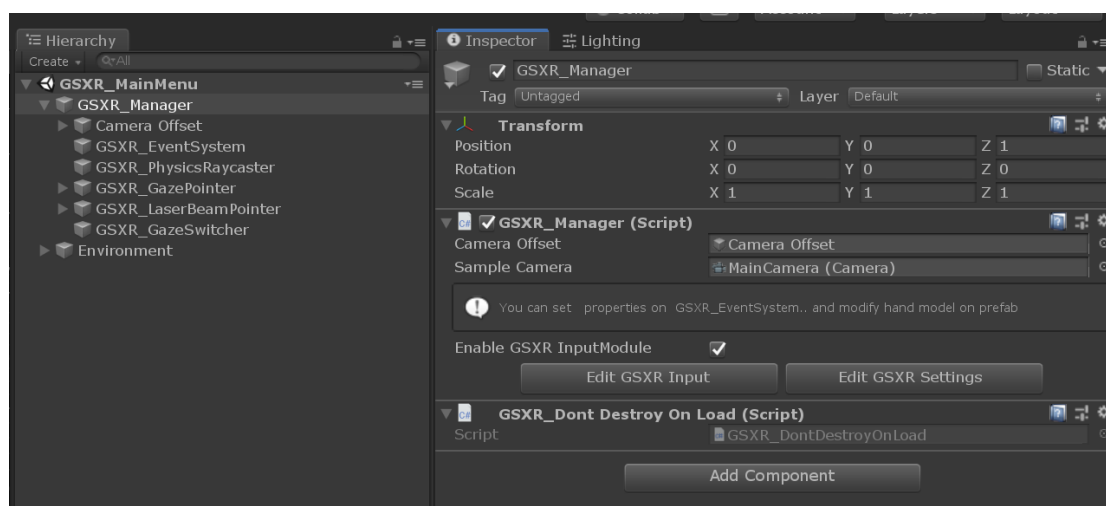


图 18 GSXR 管理器控制面板信息

4.12. 编写输入行为编辑器打开方式

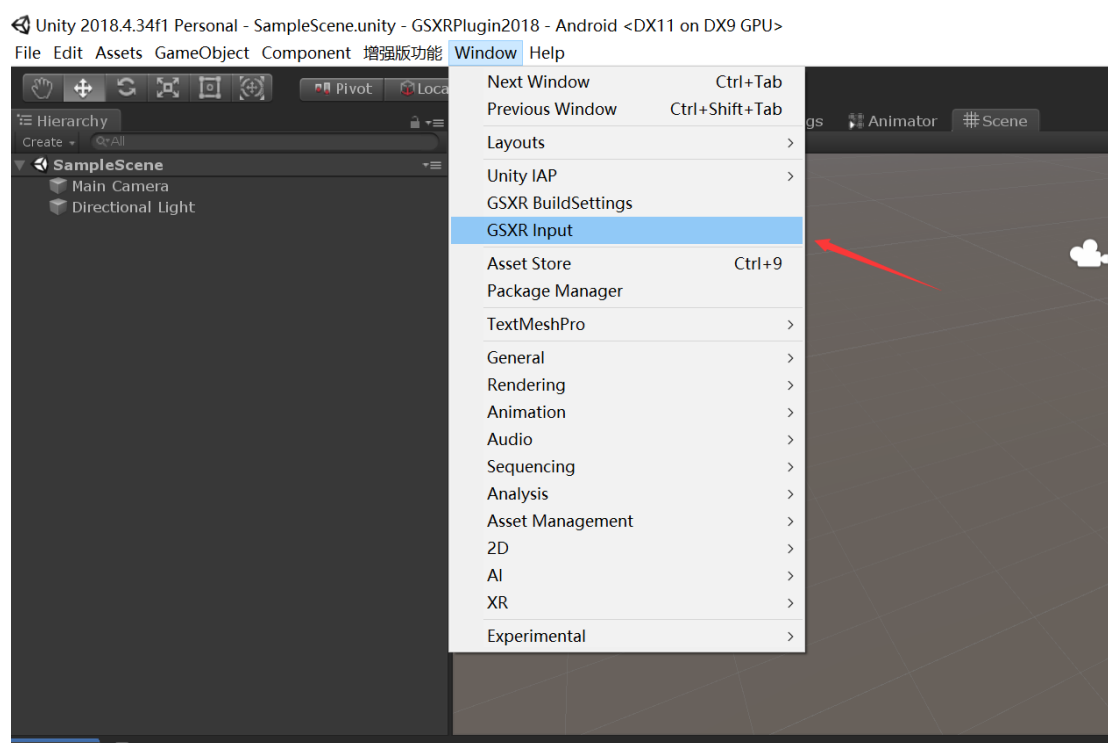


图 19 GSXR 输入行为绑定工具打开方式

4.13. 配置输入行为映射

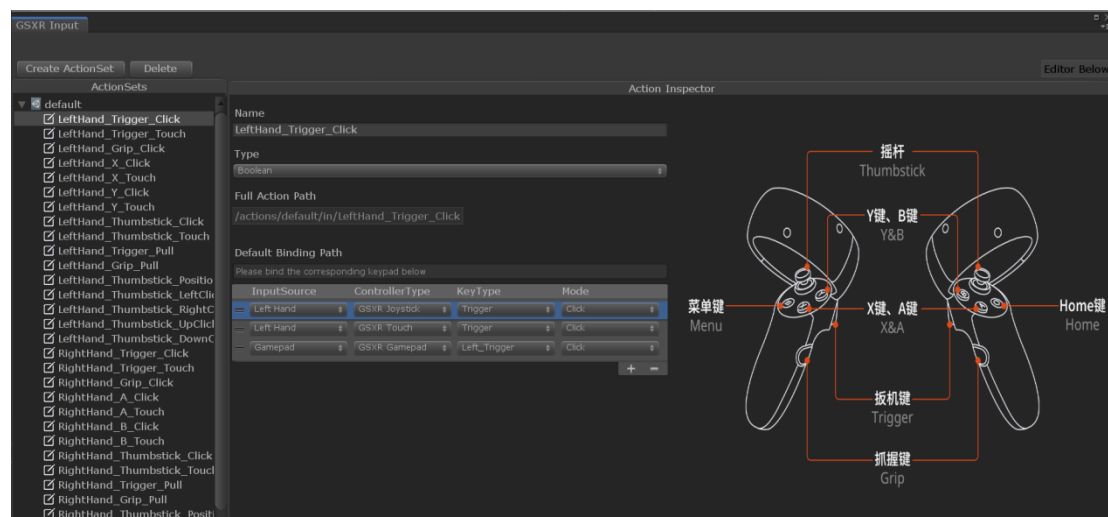


图 20 行为映射窗口

4.14. 编辑手柄预制

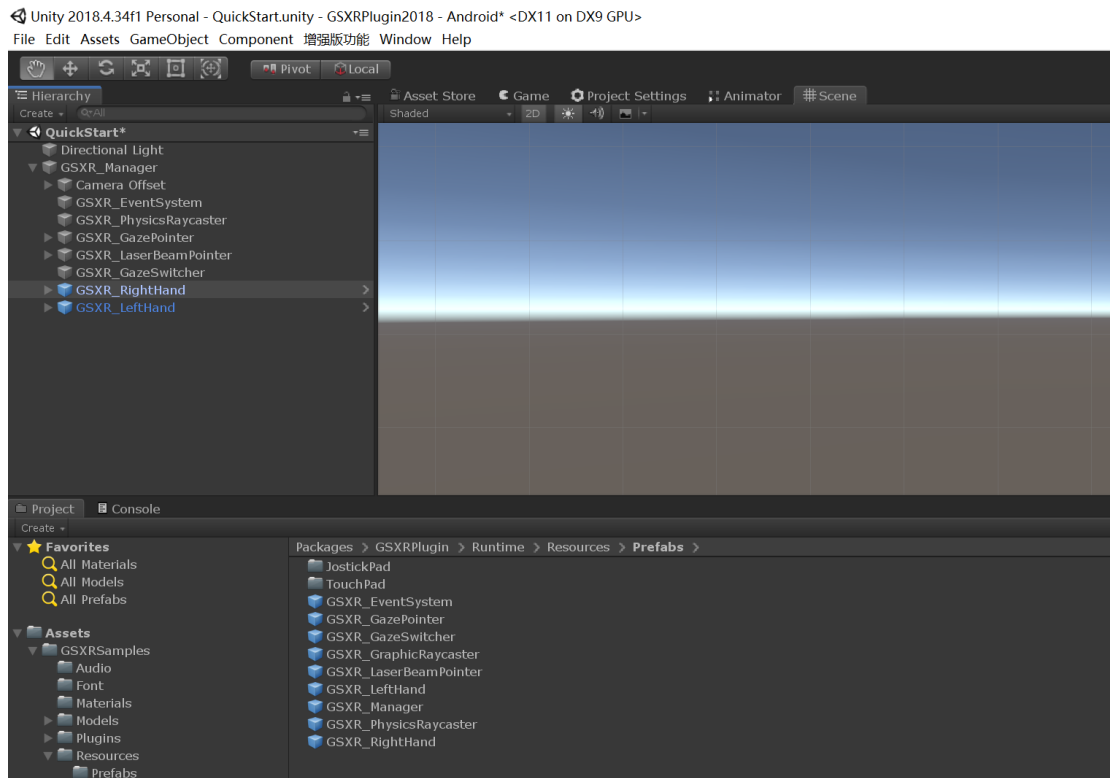


图 21 可编辑左右手柄预制体

4.15. 行为映射绑定

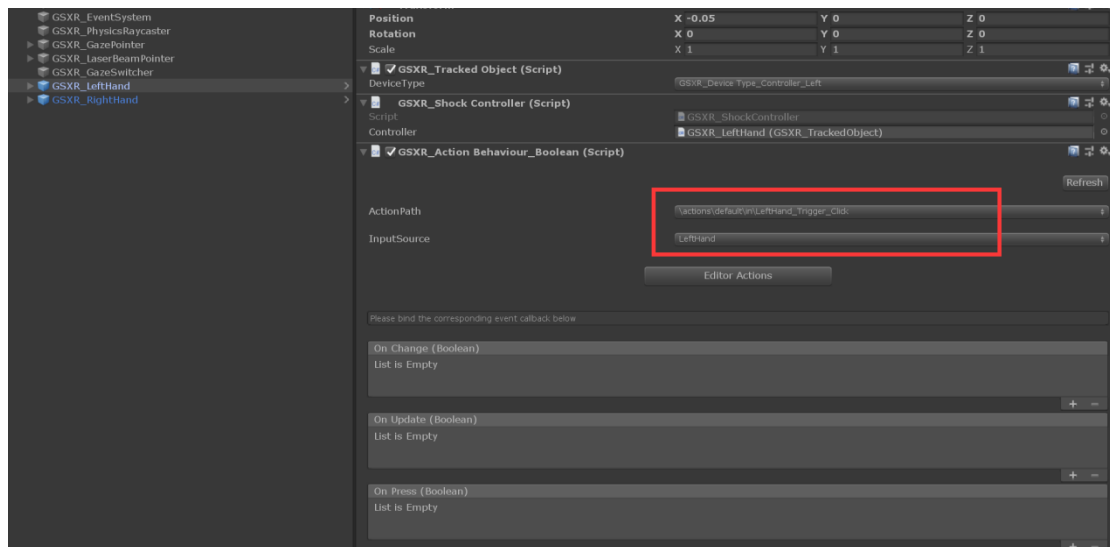


图 22 行为绑定与输入沿设置

4.16. 生成 GSXR_Settings

如果项目中没有 GSXR_Settings，可鼠标右键点击 GSXR_Settings 生成预制脚本。

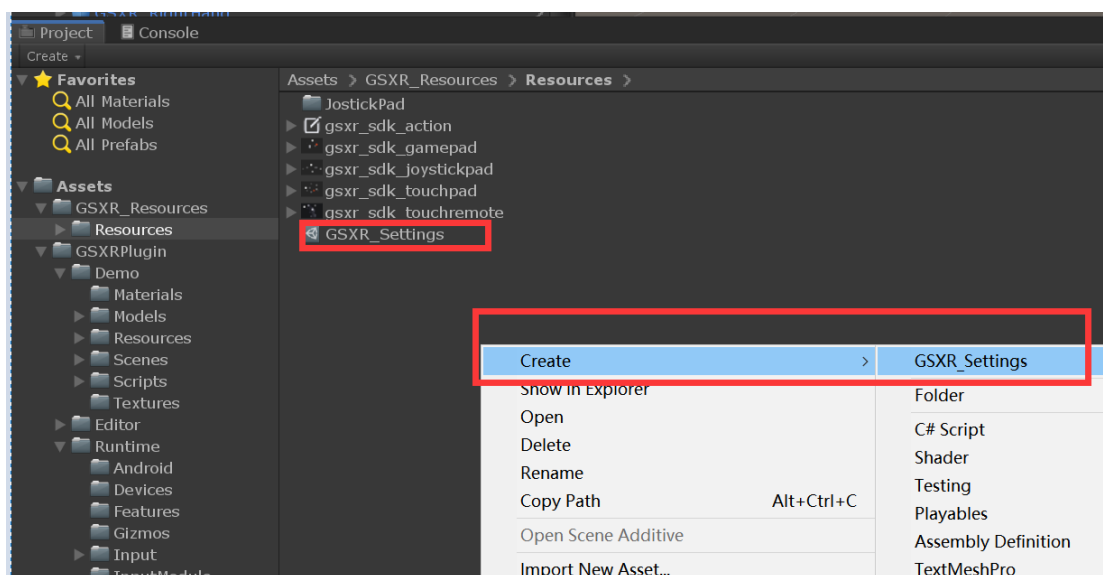


图 23 生成 GSXR_Settings

4.17. 修改 GSXR_Settings

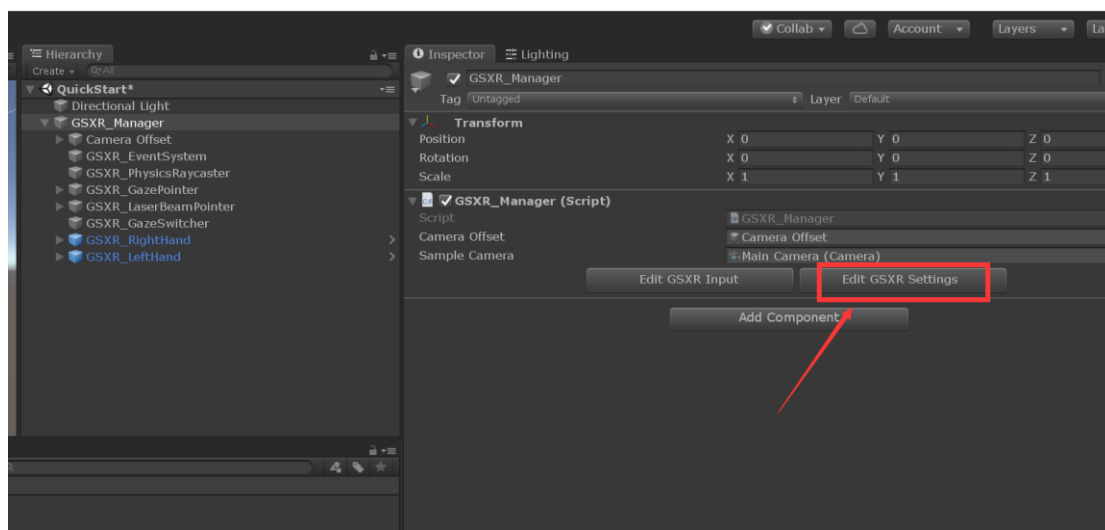


图 24 快捷寻找 GSXR_Settings

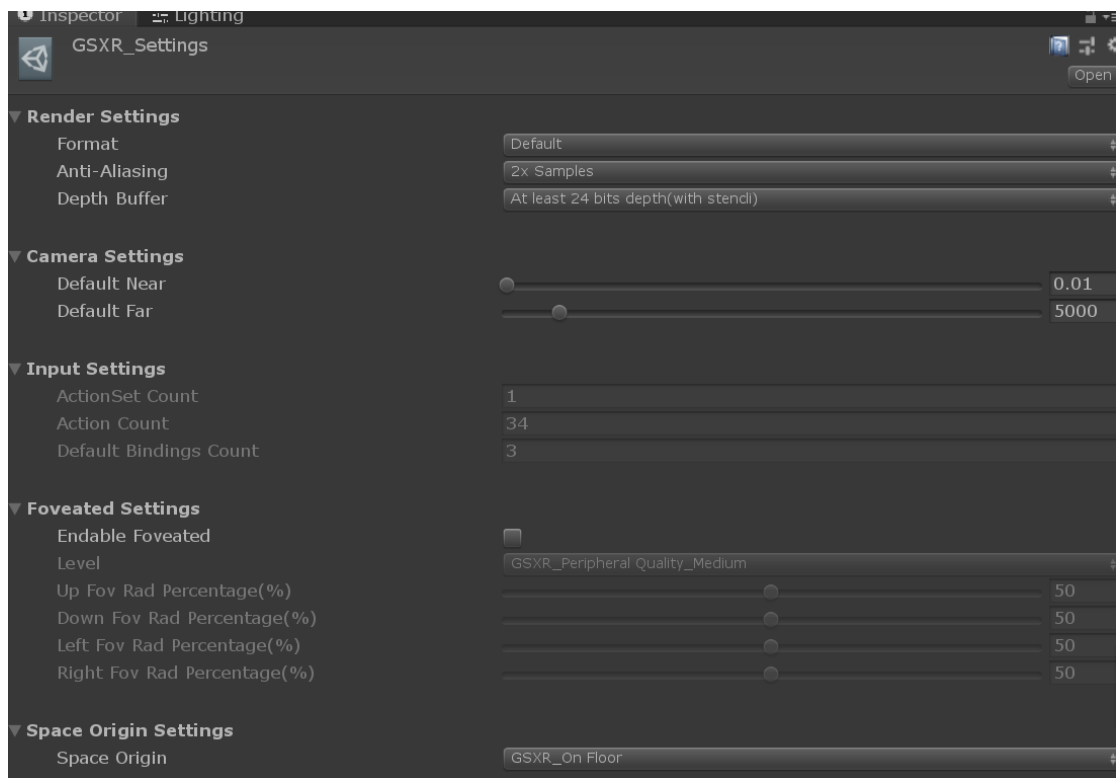


图 25 修改 GSXR_Settings

4.18. 调试运行

自定义拖入场景的左右手柄需要配置到 GSXR_GazeSwitcher 上，调试运行时会自动克隆 SampleCamera 生成对应的 GSXR_ViewSet 渲染相机；在调试模式下可通过键盘 AWS DQE 控制 ViewSet 前后左右上下，通过按 Alt+鼠标移动或者鼠标长按右键移动鼠标控制转向。

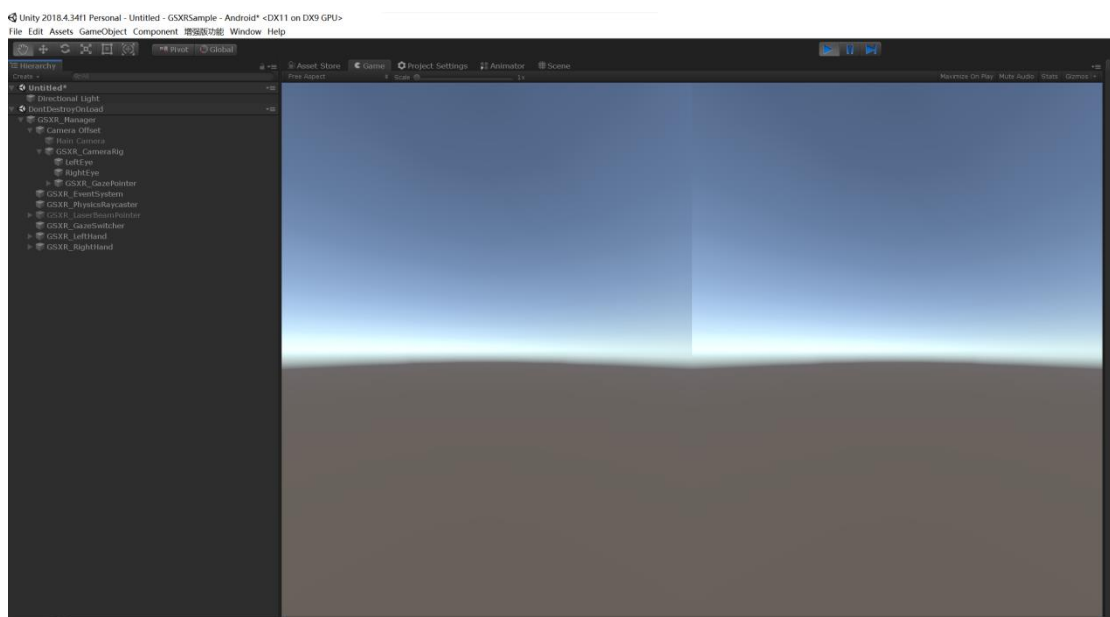


图 26 调试运行界面

4.19. 自定义添加 DontDestroyOnLoad

Enable GSXR InputModule 标志位可以切换是否使用 GSXR 事件系统与射线预制和手柄模型。

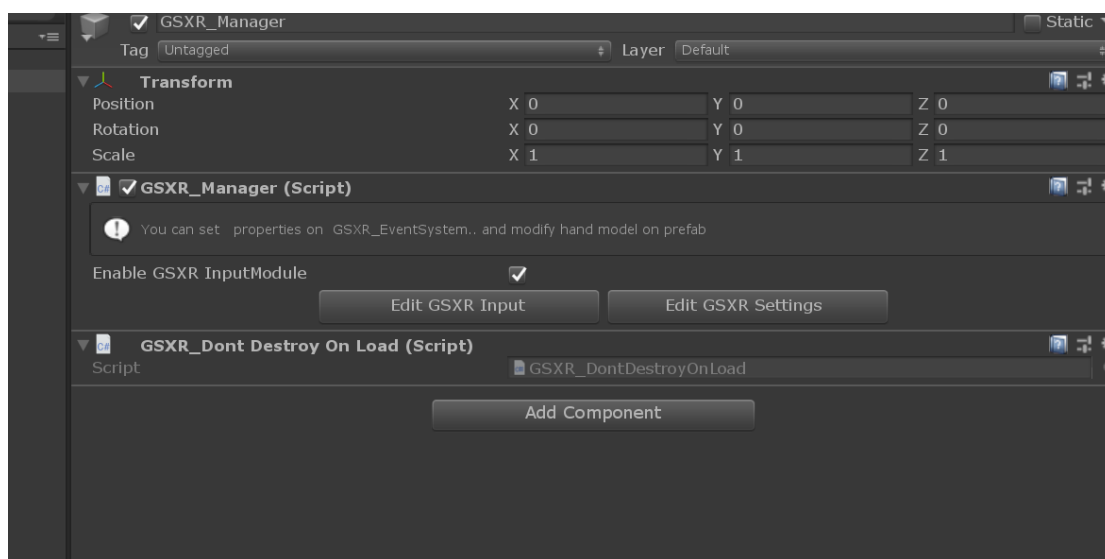


图 27 DontDestroyOnLoad 绑定

4.20. 打包设置

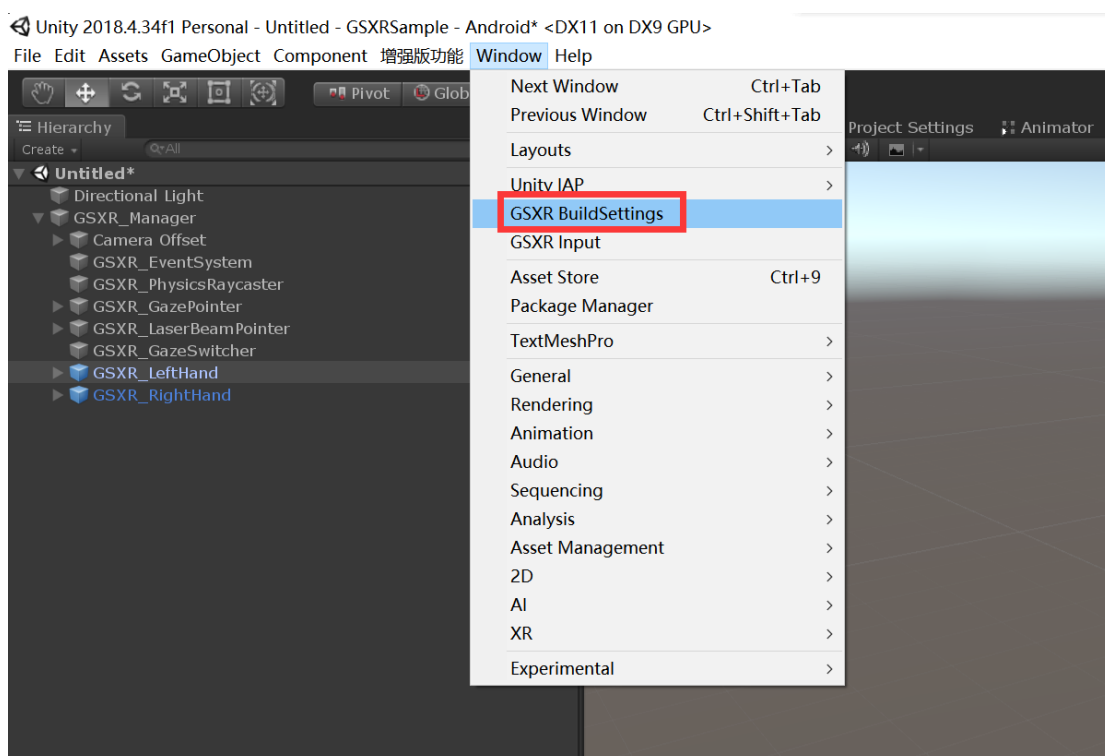


图 28 快捷打开 GSXR BuildSettings

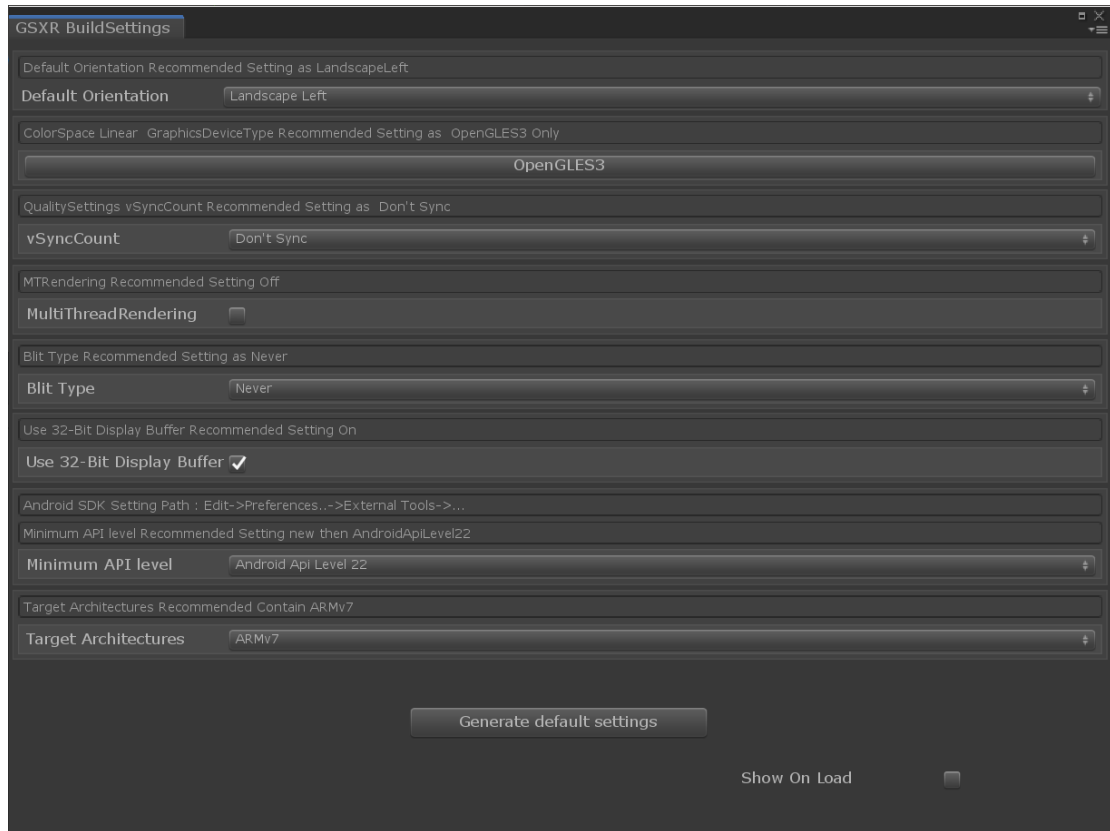


图 29 GSXR BuildSettings 推荐设置

5. 编辑行为

5.1. 行为编辑器

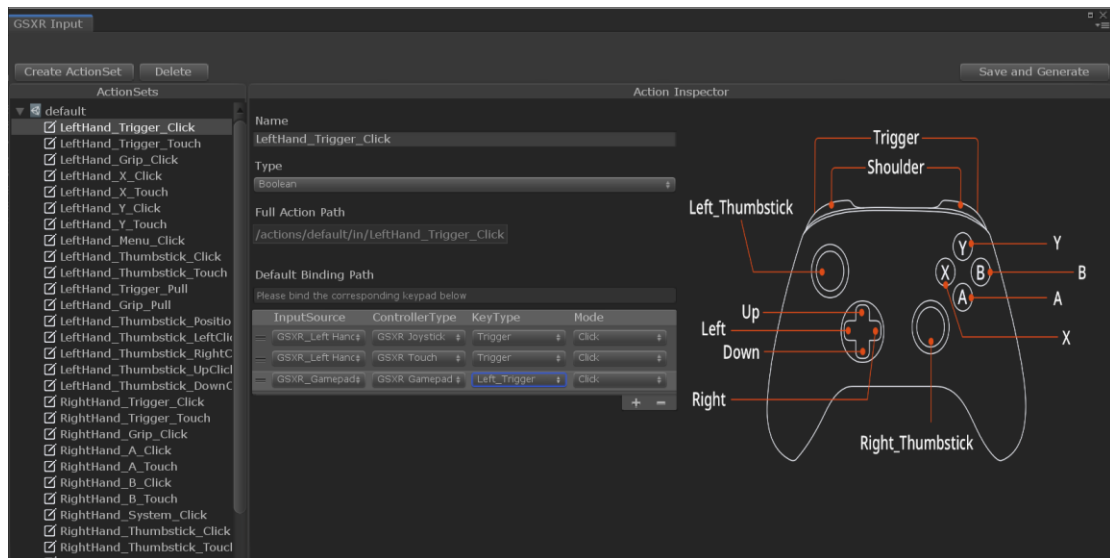


图 30 GSXR_Input 行为编辑器

现版本仅支持单个行为集编辑和修改，未来支持多个行为集编辑和修改；当选中行为集父节点时可以修改行为集的属性并添加行为；选中行为节点可以修改行为属性和编辑行为输入源绑定；编辑完成后可点击右上角保存生成按钮。

5.2. 行为绑定

5.2.1. 组件添加行为绑定集合

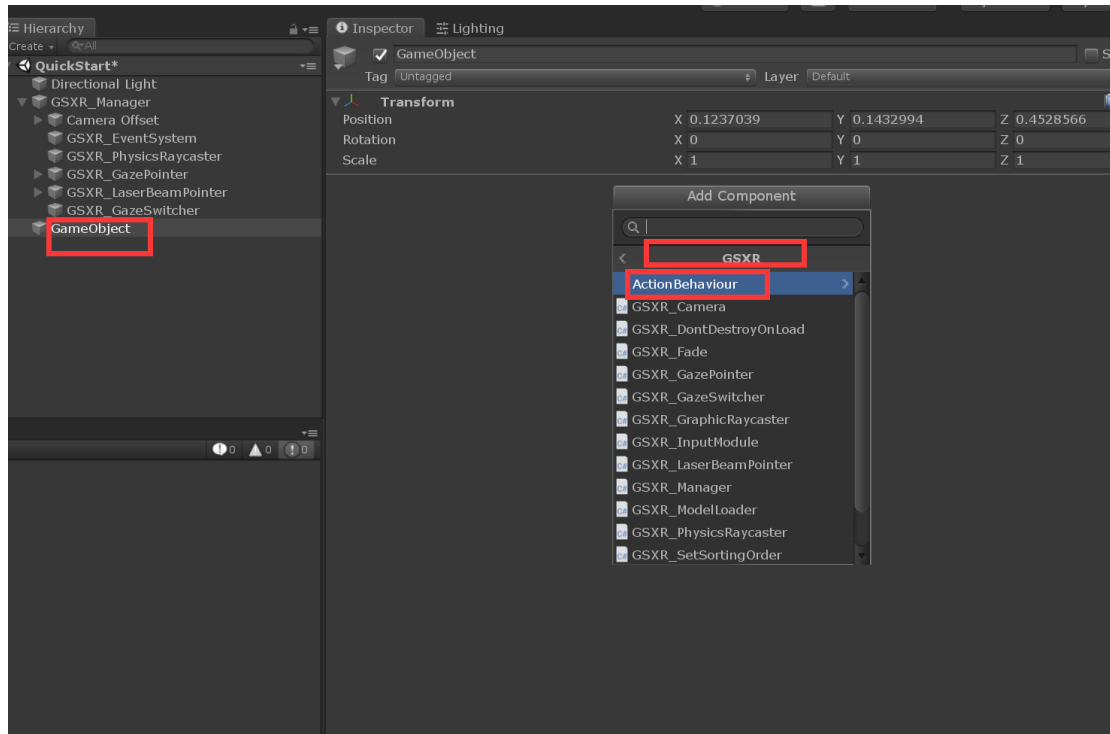


图 31 组件添加行为绑定集合

5.2.2. 组件添加行为绑定类型

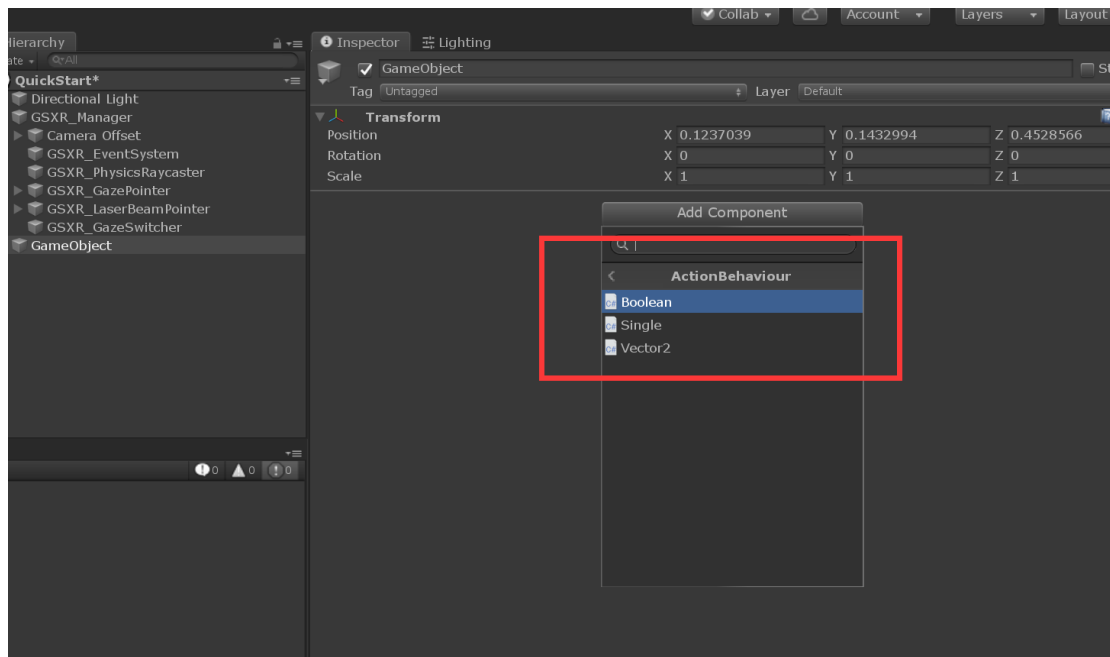


图 32 组件添加行为绑定类型

5.2.3. Boolean 行为绑定路径和输入源

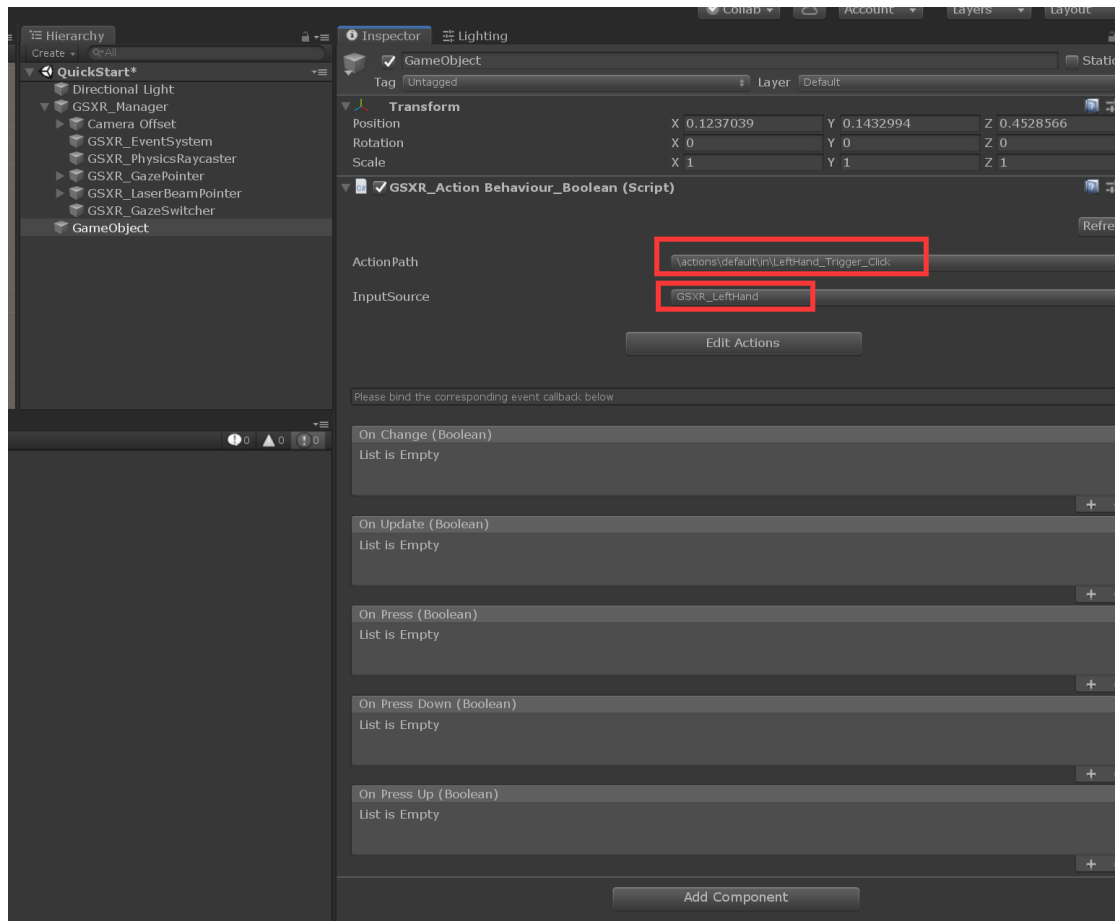


图 33 Boolean 行为绑定路径和输入源

5.2.4. Single 行为绑定路径和输入源

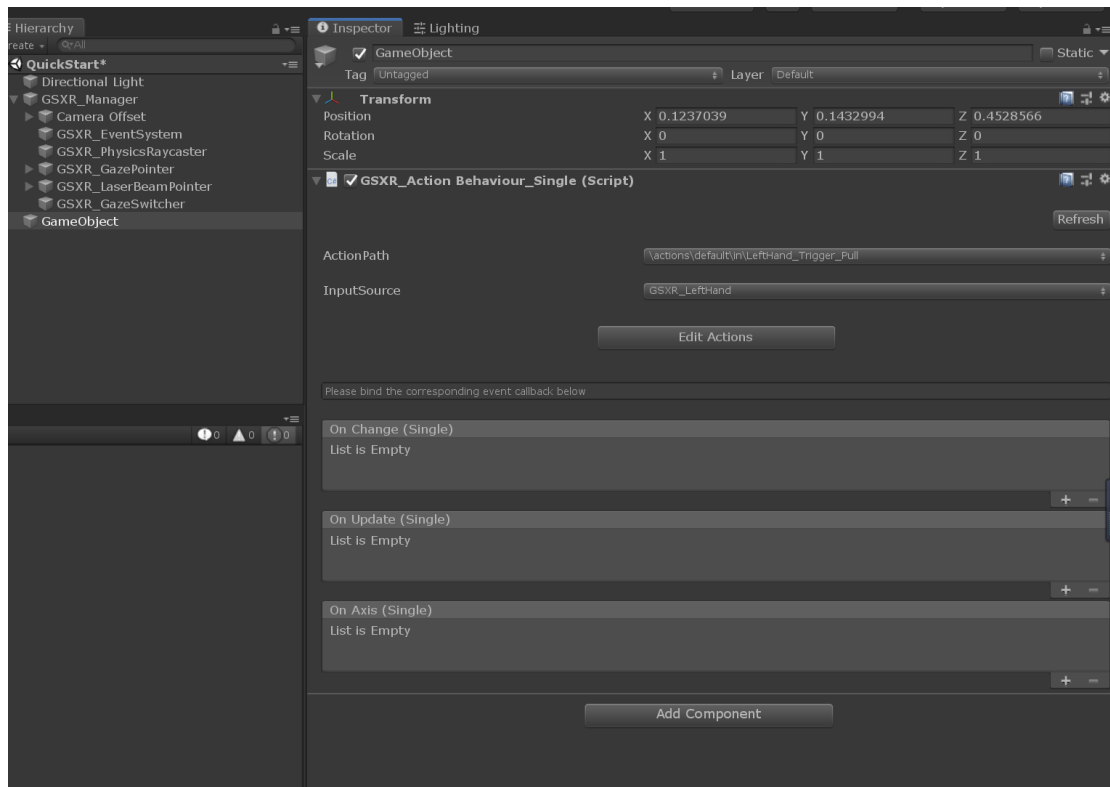


图 34 Single 行为绑定路径和输入源

5.2.5. Vector2 行为绑定路径和输入源

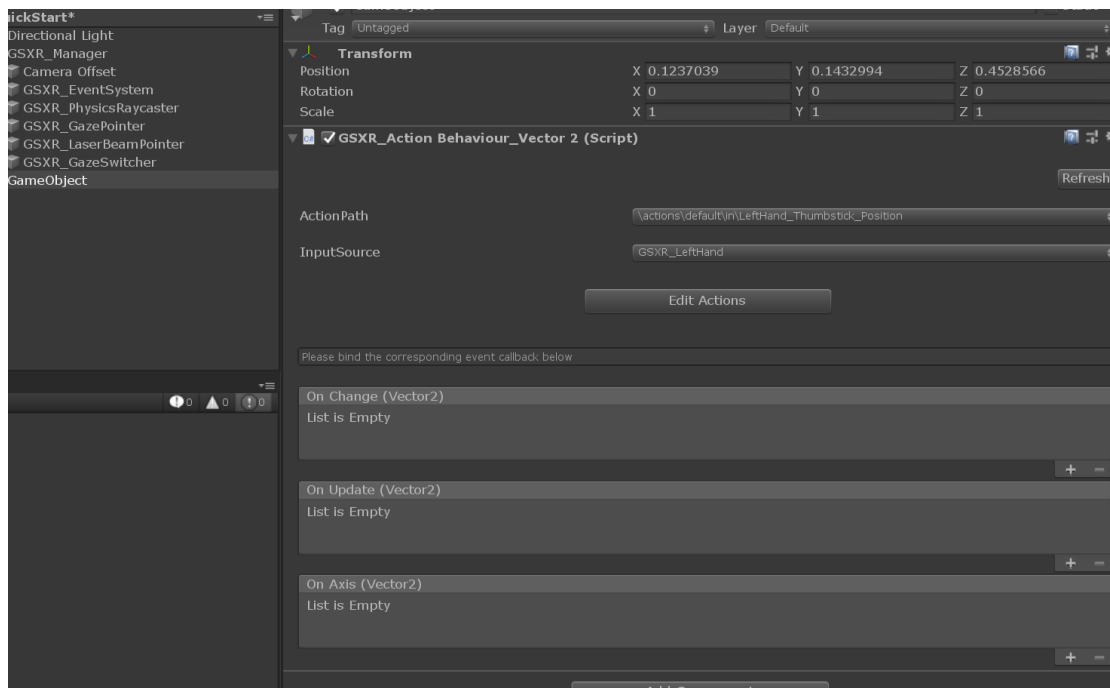


图 35 Vector2 行为绑定路径和输入源

5.2.6. 自定义脚本绑定行为路径和输入源

自定义行为脚本编写先决条件 GSXR_Input 初始化完成或者设置 GSXR_Manager 脚本执行优先级高于自定义脚本

```
// Start is called before the first frame update
void Start()
{
    if (GSXR_Input.Initialized)
    {
        LayoutGroups();
    }
    else
    {
        GSXR_Input.InitializeCompleteEvent.AddListener(LayoutGroups);
        Debug.Log("GSXR_InputActionTest InitializeCompleteEvent AddListener");
    }
}

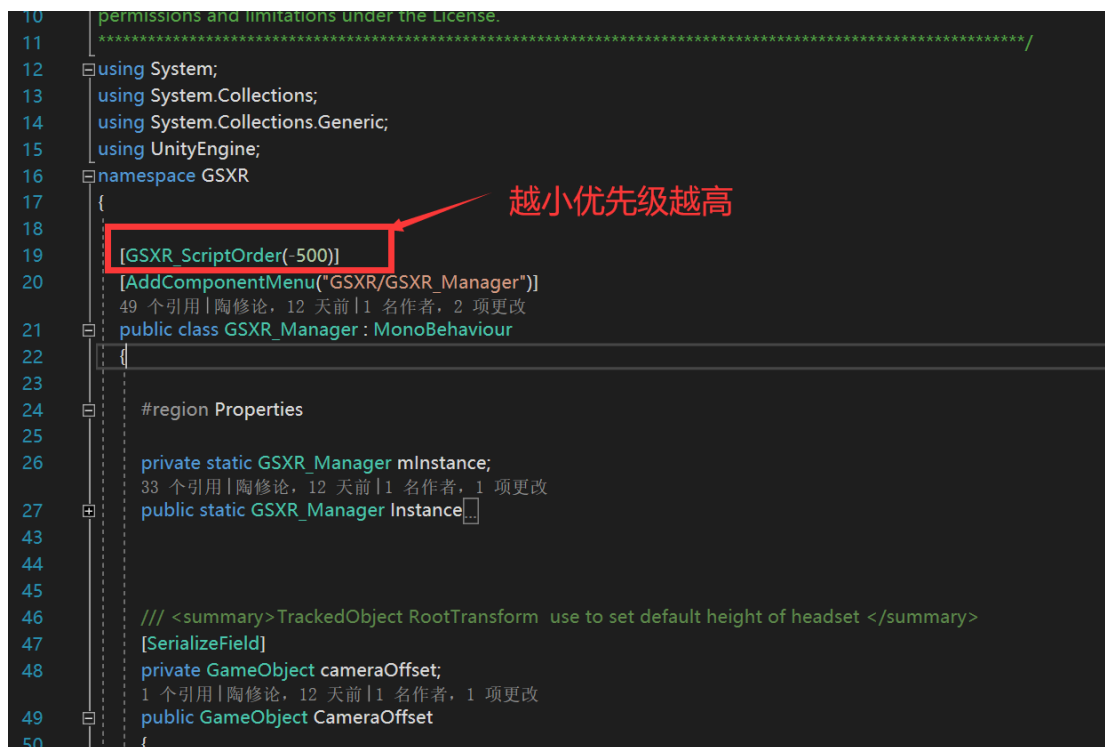
//Boolean
GSXR_Input.GetButtonDown("LeftHand_Trigger_Click", GSXR_InputSources.GSXR_LeftHand);
GSXR_Input.GetButtonDown("RightHand_Trigger_Click", GSXR_InputSources.GSXR_RightHand);
GSXR_Input.GetButtonPress("LeftHand_Trigger_Click", GSXR_InputSources.GSXR_LeftHand);
GSXR_Input.GetButtonPress("RightHand_Trigger_Click", GSXR_InputSources.GSXR_RightHand);
GSXR_Input.GetButtonUp("LeftHand_Trigger_Click", GSXR_InputSources.GSXR_LeftHand);
GSXR_Input.GetButtonUp("RightHand_Trigger_Click", GSXR_InputSources.GSXR_RightHand);
//Axis1D
GSXR_Input.GetAxis1D("RightHand_Trigger_Pull", GSXR_InputSources.GSXR_RightHand);
//Axis2D
GSXR_Input.GetAxis2D("RightHand_Thumbstick_Position", GSXR_InputSources.GSXR_RightHand);
//ActionEntity
action1 = GSXR_Input.GetBooleanAction("action1", GSXR_InputSources.GSXR_LeftHand);
action2 = GSXR_Input.GetSingleAction("action2", GSXR_InputSources.GSXR_LeftHand);
action3 = GSXR_Input.GetVector2Action("action3", GSXR_InputSources.GSXR_LeftHand);
```

5.2.7. 获取 Hmd 键值

```
//Boolean
GSXR_Input.GetHmdKeyDown( GSXR_Hmd_InputId.GSXR_Hmd_InputId_Enter);
GSXR_Input.GetHmdKeyUp(GSXR_Hmd_InputId.GSXR_Hmd_InputId_Enter);
GSXR_Input.GetHmdKeyPress(GSXR_Hmd_InputId.GSXR_Hmd_InputId_Enter);

//...
```

5.2.8. 设置 GSXR_Manager 脚本执行优先级



```
10 permissions and limitations under the License.
11 *****/
12 using System;
13 using System.Collections;
14 using System.Collections.Generic;
15 using UnityEngine;
16 namespace GSXR
17 {
18     [GSXR_ScriptOrder(-500)]
19     [AddComponentMenu("GSXR/GSXR_Manager")]
20     49 个引用 | 陶修论, 12 天前 | 1 名作者, 2 项更改
21     public class GSXR_Manager : MonoBehaviour
22     {
23
24         #region Properties
25
26         private static GSXR_Manager mInstance;
27         33 个引用 | 陶修论, 12 天前 | 1 名作者, 1 项更改
28         public static GSXR_Manager Instance
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46     /// <summary>TrackedObject RootTransform use to set default height of headset </summary>
47     [SerializeField]
48     private GameObject cameraOffset;
49     1 个引用 | 陶修论, 12 天前 | 1 名作者, 1 项更改
50     public GameObject CameraOffset
51     {
52         get { return cameraOffset; }
53         set { cameraOffset = value; }
54     }
55 }
```

图 36 自定义脚本优先级

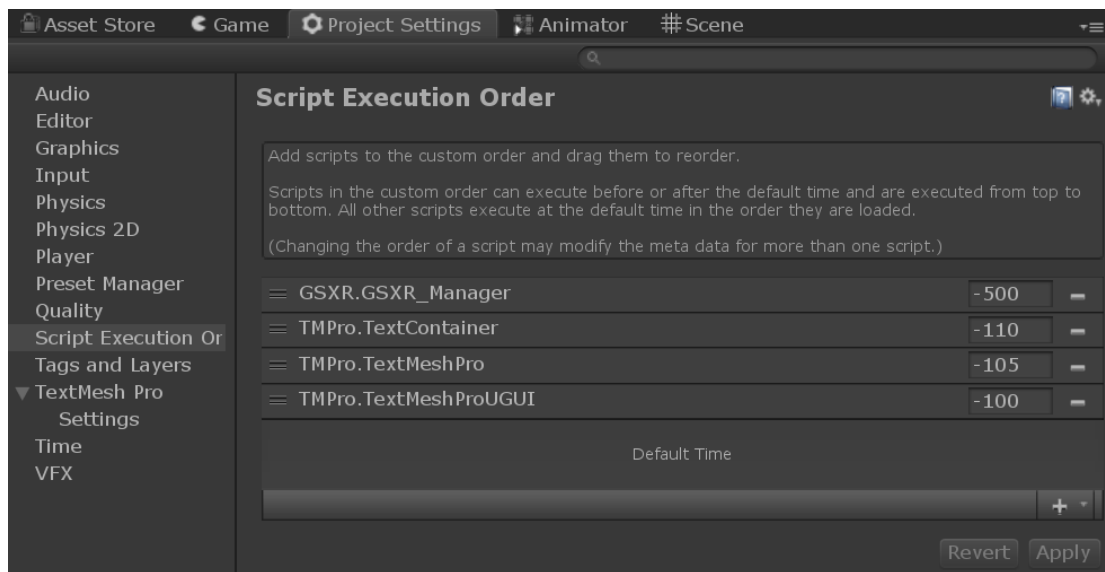


图 37 查看自定义脚本优先级

6. 全局设置

6.1. 设置文件路径

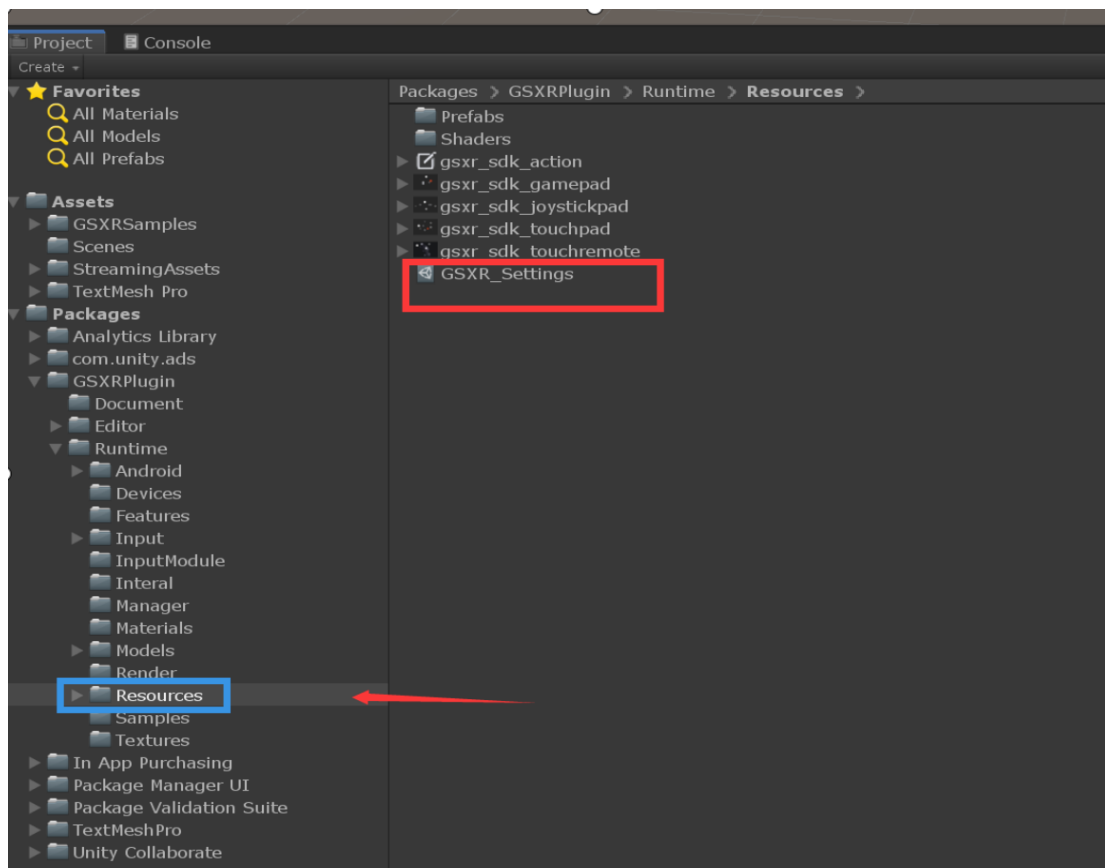


图 38 GSXR_Settings.asset 文件路径

6.2. 设置快捷指引

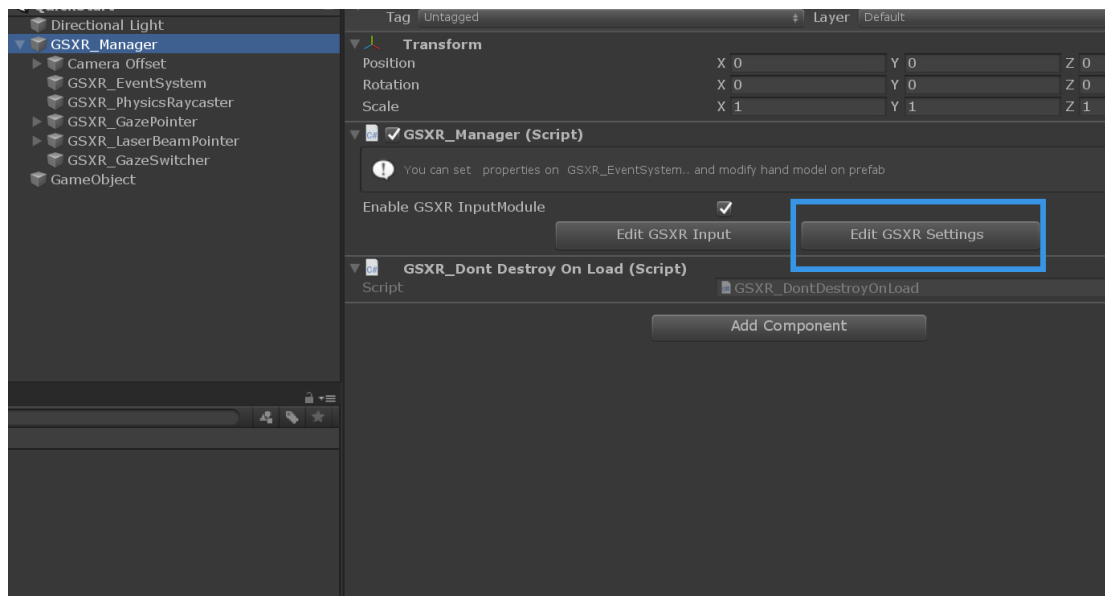


图 39 GSXR_Manager 管理器快捷进入全局设置按钮

6.3. 设置参数说明

6.3.1. RenderSettings

传输纹理格式默认设为 Default ,可自定义修改抗锯齿参数和图像深度。
Anti-Aliasing 参数仅供设置当前视图集 RenderTexture 的质量,如果需要修改抗锯齿等全局质量请到 ProjectSettings/Quality 下根据不同平台自定义修改。

6.3.2. CameraSettings

相机近裁切面和远裁切面的距离;当 SampleCamera 的参数 Near > Default Near 时使用 Default Near 的值;当 SampleCamera 的参数 Far < Default Far 时使用 Default Far 的值。

6.3.3. InputSettings

展示 Input 行为集合的数目,当数目为 0 或者 没有显示参数时 则需要打开 GSXR_Input 编辑器配置。

6.3.4. FoveatedSettings

注视点渲染,使能后可以设置参数。

6.3.5. SpaceOriginSettings

空间原点设置，GSXR_OnHead 原点始于头戴设备本身，GSXR_OnFloor 原点考虑身高，校准至地板上。

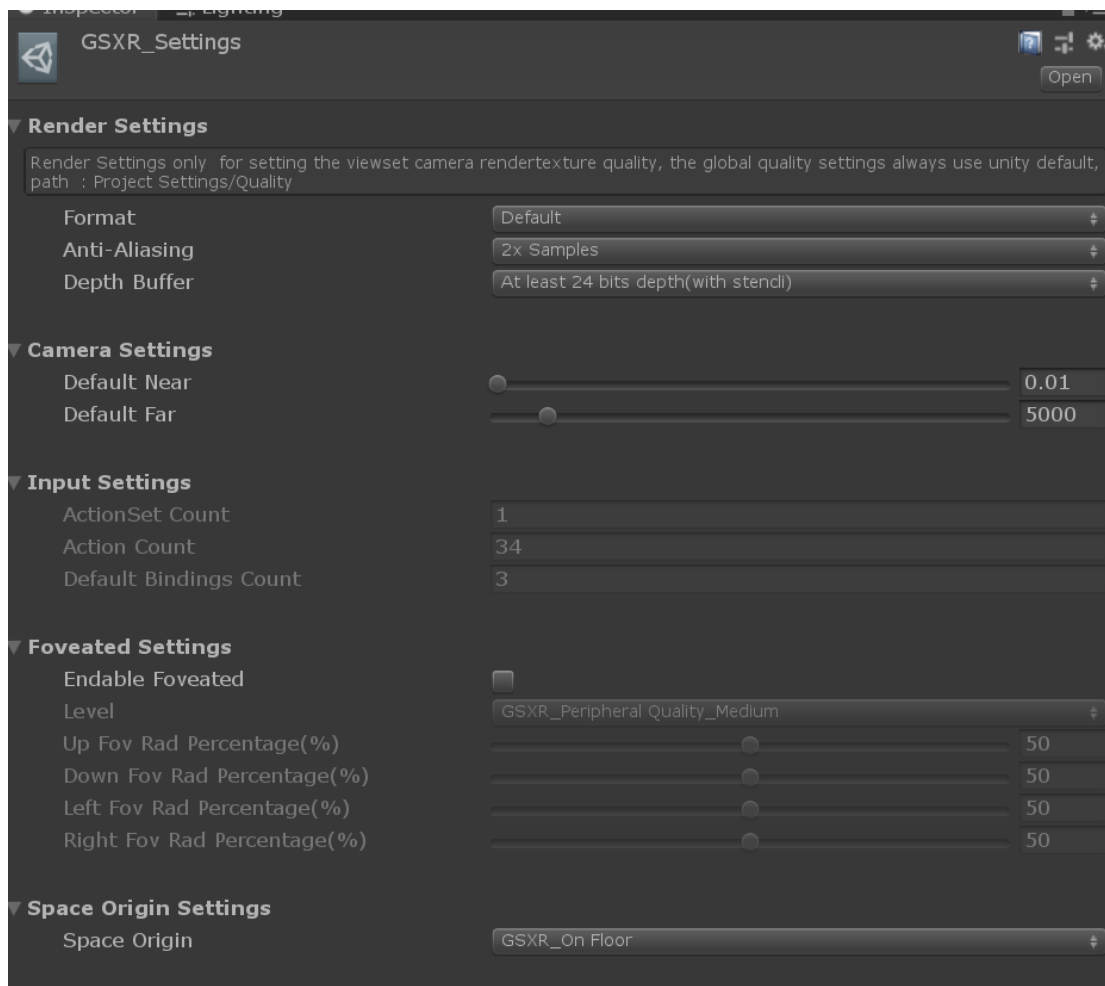


图 40 GSXR_Setting 全局设置界面

7. 异常处理

7.1. 版本异常

解决办法：使用支持的版本

7.2. URP/LWRP 模板导入后 dll 拷贝失败

解决办法：重启项目

8. 注意事项

8.1. 设计返回逻辑

GSXR 内容上线 GSXR 平台需设计对应的返回键退出应用逻辑，如在业务设计中存在业务返回逻辑，当业务处于最外层时退出应用。

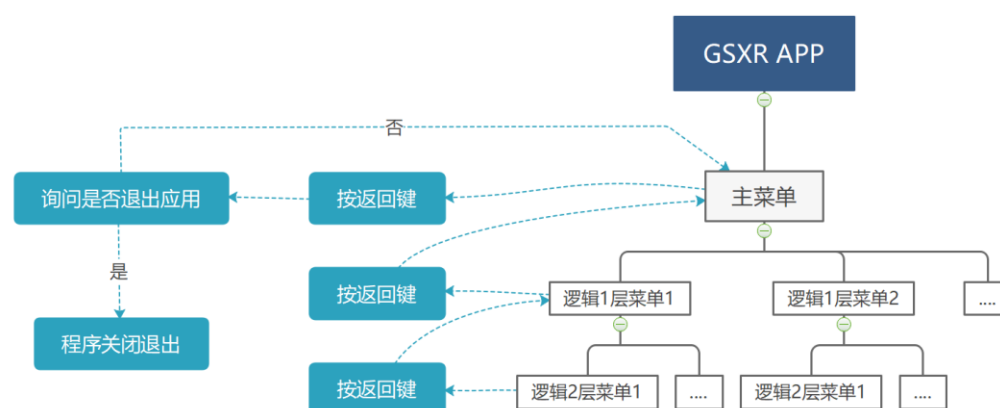


图 41 GSXR 应用返回逻辑参考图