HUAWEI VR SDK 3.0 Developer Guide

Issue: V3.0.0

Trademarks and Permissions

and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Change History

Date	Version	Description
2019-11-16	V3.0.0	Added the adaptation guide for HUAWEI VR Controller prefabs.

Contents

1 Overview

1.1 Function Introduction

You can use the HUAWEI VR SDK 3.0 and Unity to develop HUAWEI VR apps. HUAWEI VR SDK 3.0 for Unity (SDK 3.0 for short) provides the following functions:

- App scene customization.

  API functions can be used to replace the skybox materials, for quick transitions between the different app scenes.

- Gyro sensor data capture.

  HUAWEI VR glasses provide for more accurate gyro sensor data, which is obtained in real time through the API functions.

- Supports controller operations.

  The controller interaction function has been integrated. Through the API functions, you can obtain the controller status, type, posture, and touch panel and button statuses; reset controller posture; and preset shortcuts.

- Supports phone-controller interactions.

  Phone-controller interactions have been integrated. You can replace certain interactive controller functions with operations on the phone screen, without any additional development or adaptation required.

- Volume adjustments.

  The volume adjustment function has been incorporated into VR mode. The volume and volume display can be adjusted through the buttons on the HUAWEI VR Controller or the wire control, without any additional development or adaptation required.

- Call answering, rejecting, and ending.

  The answering, rejecting, and ending call functions have been integrated, without additional development or adaptation required. When a call comes in, the incoming call mode is enabled. Users can then answer, reject, or hang up the call as prompted.

- Automatic notification message display.

  The automatic notification dialog box display function has been integrated, without any additional development or adaptation required. When a notification message is received, a dialog box displays. The notification message types are as follows: SMS message dialog box, general message dialog box, low-battery warning dialog box, Wi-Fi connection change dialog box, and temperature alarm dialog box.

- Settings screen.

  The settings function has been integrated into VR mode, including the Do not disturb mode and brightness adjustment capabilities, without any additional development or adaptation required.

- Locking or unlocking the angle of view.

  In VR mode, the angle of view in a VR scene can be locked or unlocked, through use of the API functions.

- Scene-specific camera rendering resolution.

You can modify the camera rendering resolution by setting the various scale factors.

- 3D sound effects.

  Provides audio plug-ins to apply 3D sound effects. You can implement sound direction rendering, distance simulation, and environmental sound effect rendering in various 3D virtual scenes, to achieve a highly-immersive audio experience.

- Opening in 2D.

  You can open the VR app in a 2D interface, after special configurations. For details, please see section 2.3.4 in the developer guide.

- In-app screenshots.

  Screenshot can be taken by pressing the Home and Trigger buttons together, when using the app, without any additional development or adaptation required.

- Adaptive resolution.

  Automatic resolution adaption has been integrated into VR mode, without requiring additional development or adaptation. When the phone temperature is too high, the resolution will be reduced automatically, in order to preserve power.

- Forward compatibility.

  Apps developed in SDK 3.0 can also be used on HUAWEI VR 2.0.

- Touch panel track points display.

  In VR mode, touch positions on the touch panel are displayed, without any additional development or adaptation required.

- Controller tooltip display.

  The controller tooltip display function has been integrated, without any additional development or adaptation required. When the controller is close to the VR glasses, the tooltip for the controller buttons and shortcuts will automatically display. You can also customize the message to be displayed.

1.2 Basic Knowledge Requirements

Before using the SDK 3.0, you need to have a basic understanding of the Unity platform and C# language.

1.3 Supported Platforms

- Smartphone: Huawei flagship smartphones, such as the Mate 30 series, Mate 20 series, P30 series, and P20 series.
- Glasses: HUAWEI VR 2 and HUAWEI VR Glass.
- Operating system: EMUI 8.1 or later.
- Unity version: 5.6.x to 2018.2.x

2 Development Guide

It is recommended that you use the LTS version of Unity 5.6.x to 2018.2.x to build the HUAWEI VR apps on Android. This document will walk you through the Unity configuration, development, and HUAWEI VR app development.

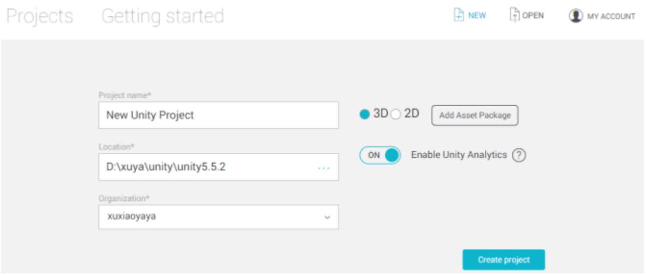
2.1 Preparations

- Install the Unity platform on the PC.
- Download the **hvrsdk-unity-3.0.0.x** package.
- Install the HUAWEI VR SDK service APK on the phone.
- Get the HUAWEI phone and HUAWEI VR glasses ready.

2.2 Development Procedure

2.2.1 Creating a Unity Project and Importing SDK 3.0

- Open Unity and create a 3D project, as shown in Figure 2-1.



Project name*: New Unity Project

Location*: D:\xuya\unity\unity5.5.2

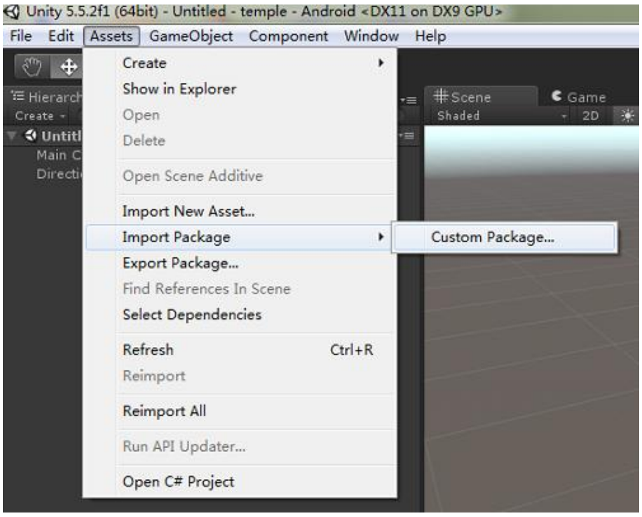
Organization*: xuxiaoyaya

● 3D ○ 2D

ON Enable Unity Analytics

Figure 2-1 Unity startup screen

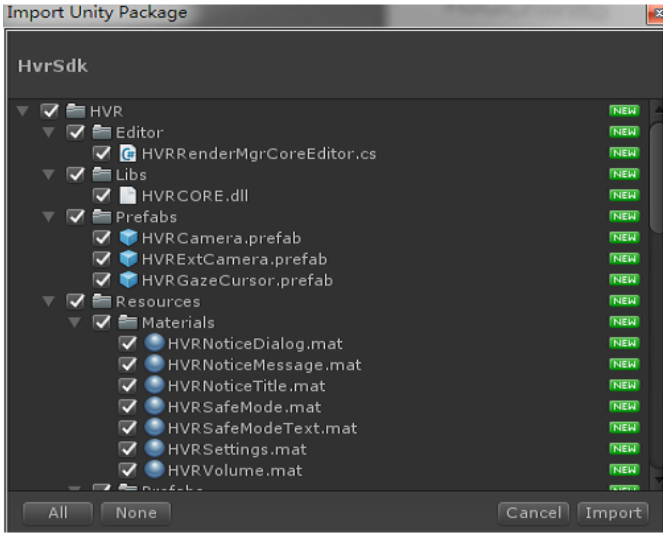
- Go to **Assets** >**Import Package** > **Custom Package**, as shown in Figure 2-2.



Unity 5.5.2f1 (64bit) - Untitled - temple - Android <DX11 on DX9 GPU>

Figure 2-2 Screen for importing **unitypackage**

- In the displayed dialog box, select the downloaded **hvrsdk-3.0.0.x.unitypackage** and click **Import**, as shown in Figure 3.



Import Unity Package

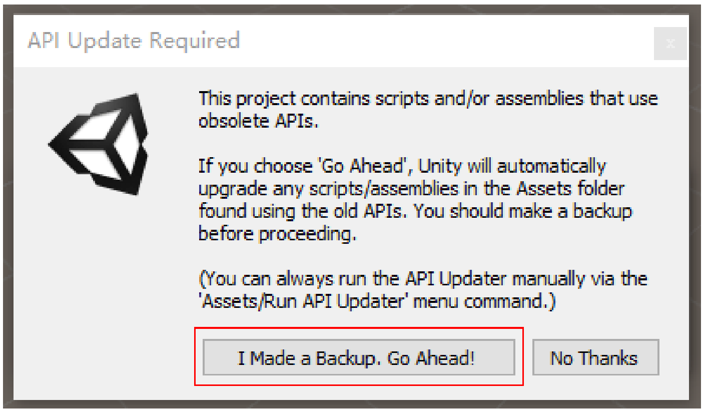
HvrSdk

- ☑ HVR NEW
  - ☑ Editor NEW
    - ☑ HVRRenderMgrCoreEditor.cs NEW
  - ☑ Libs NEW
    - ☑ HVRCORE.dll NEW
  - ☑ Prefabs NEW
    - ☑ HVRCamera.prefab NEW
    - ☑ HVRExtCamera.prefab NEW
    - ☑ HVRGazeCursor.prefab NEW
  - ☑ Resources NEW
    - ☑ Materials NEW
      - ☑ HVRNoticeDialog.mat NEW
      - ☑ HVRNoticeMessage.mat NEW
      - ☑ HVRNoticeTitle.mat NEW
      - ☑ HVRSafeMode.mat NEW
      - ☑ HVRSafeModeText.mat NEW
      - ☑ HVRSettings.mat NEW
      - ☑ HVRVolume.mat NEW

Figure 2-3 Unitypackage directory

NOTE

- ➢ If the following dialog box is displayed during the import, click **I Made a Backup, Go Ahead!**, as shown in Figure 2-4. Otherwise, the compiled app may fail to find APIs.



API Update Required

This project contains scripts and/or assemblies that use obsolete APIs.

If you choose 'Go Ahead', Unity will automatically upgrade any scripts/assemblies in the Assets folder found using the old APIs. You should make a backup before proceeding.

(You can always run the API Updater manually via the 'Assets/Run API Updater' menu command.)

Figure 2-4 Update dialog box

- ➢ After the **HvrSdk.unitypackage** file is imported, **Assets/HVR/Scripts/HVRAddUILayer.cs** automatically adds two UI Layers, the **HVRUILayer** and **HVRVolumeUILayer**, for drawing of specific UI and volume dialog box respectively, as shown in Figure 2-5. If the UI layers fail to be automatically added, a dialog box is displayed, as shown in Figure 2-6. You will need to manually add the UI Layers named **HVRUILayer** and **HVRVolumeUILayer**.

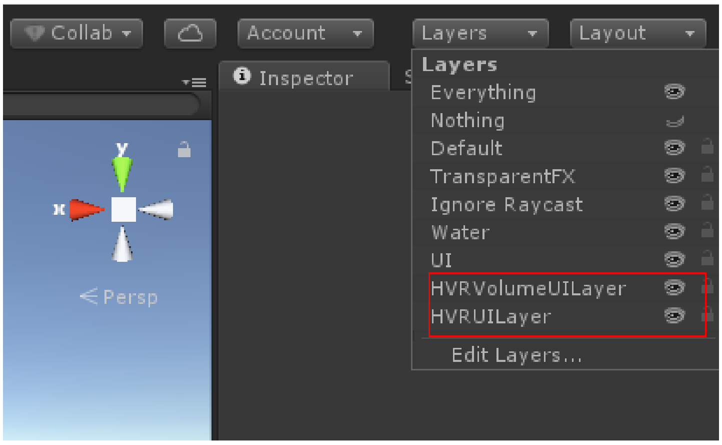
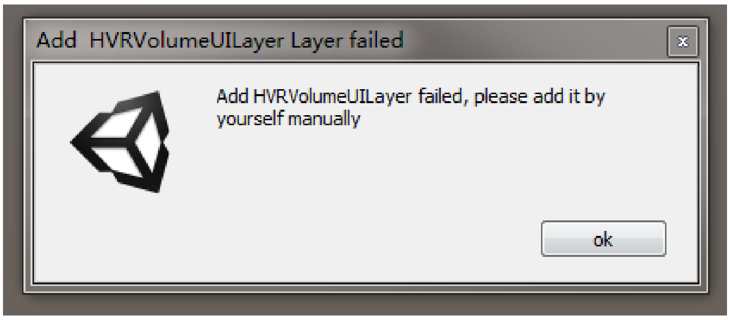


Figure 2-5 Layers screen


Add HVRVolumeUILayer Layer failed

Add HVRVolumeUILayer failed, please add it by yourself manually

Figure 2-6 Layer adding failure

- Do not draw other contents on the two UI layers. In addition, the two UI layers are not displayed for other cameras in the Unity scene. For example, after Culling Mask is modified for the left and right cameras of SDK 3.0, the contents of the two UI layers are displayed.
- If you want to modify the Culling Mask of the HUAWEI VR Camera, make sure that the camera does not draw the two UI Layers, for example:

```
HVRLayoutCore.m_LeftCamObj.GetComponent<Camera>().cullingMask &= ~(1 <<
LayerMask.NameToLayer(HVRDefCore. m_VolumeUILayerName));)
```

2.2.2 Adding a HUAWEI VR Camera in a Scene

The HUAWEI VR camera needs to be added into the scene for the scene to be displayed in VR mode. Drag **HVRCamera** in the **Assets/HVR/Prefabs** directory and drop it in the **Hierarchy** workspace, and then delete the **MainCamera** automatically generated when a Unity project is created, as shown in Figure 2-7.

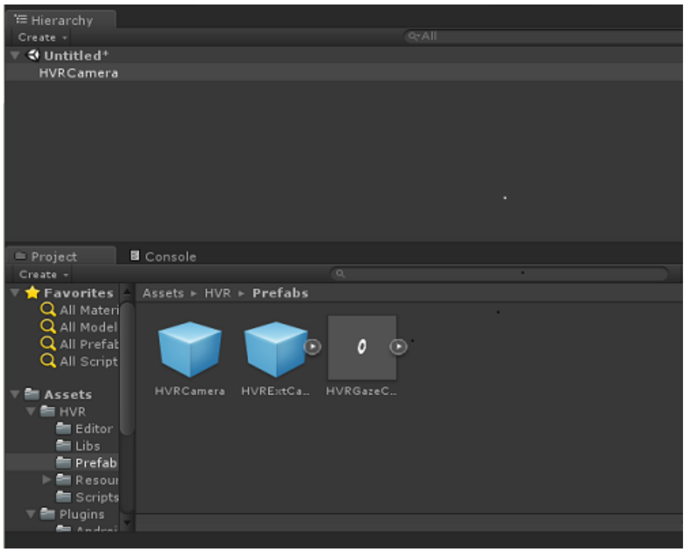


Figure 2-7 Adding HVRCamera

After **HVRCamera** is added, the VR mode can be displayed on the mobile phone. Click **HVRCamera**, and the attributes are displayed in the **Inspector** area, as shown in Figure 2-8.

➢ **Render Depth Format** is the depth parameter when **RenderTexture** is initialized in Unity. The default value is 24 Bit. You can set it to 16 Bit, 24 Bit, or 32 Bit.

➢ **Render Resolution Scale** is the resolution scaling multiple when **RenderTexture** is initialized in Unity. The default value is 1, and the default resolution is 1080 x 1080. Developers can set the rendering resolution based on the scene complexity.

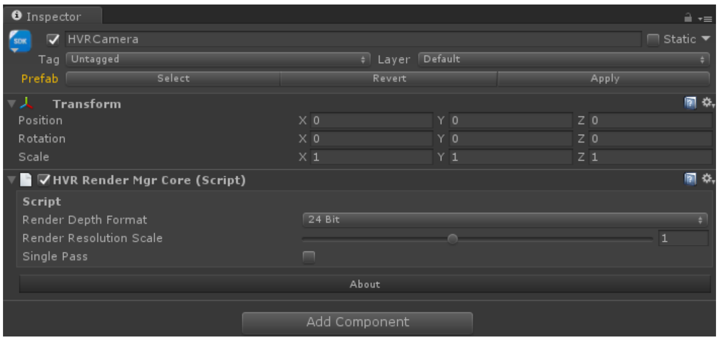
NOTE

- Modifying the resolution changes the content definition and affects the performance and power consumption.
- By default, the SDK sets and adjusts the resolution based on the temperature control solution. The resolution adjusted by **Render Resolution Scale** takes effect only when the temperature control solution is not functioning.

➢ **Single Pass** is enabled only when Multiview mode is enabled. For details about how to enable Multiview mode, see section 0.

NOTE

Do not select **Single Pass** in a mode other than Multiview. Otherwise, the display may be faulty.



☑ HVRCamera
Tag Untagged Layer Default
Prefab

Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

☑ HVR Render Mgr Core (Script)
Script
Render Depth Format 24 Bit
Render Resolution Scale 1
Single Pass □

Figure 2-8 Rendering and resolution configurations

2.2.3 Adding a HUAWEI VR Controller in a Scene

After adding a VR camera, drag the HUAWEI VR controller prefabs, the **HVRController** and **HVREventSystem** in the **Assets/HVRController/Prefabs** directory, to the scene, as shown in Figure 2-9.

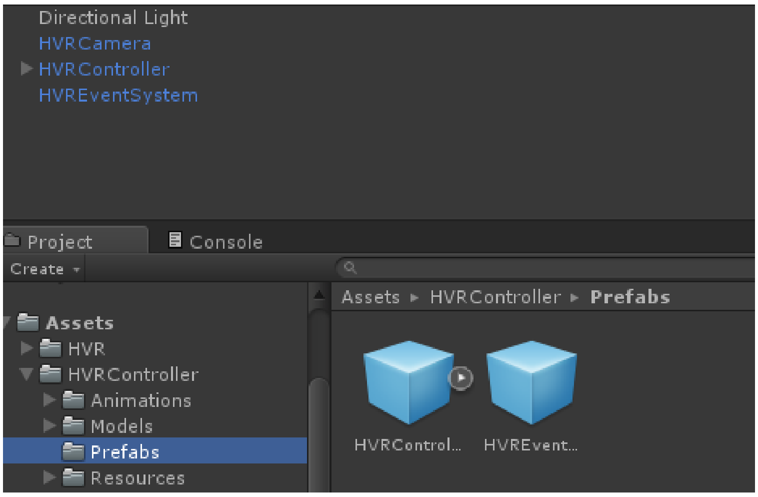
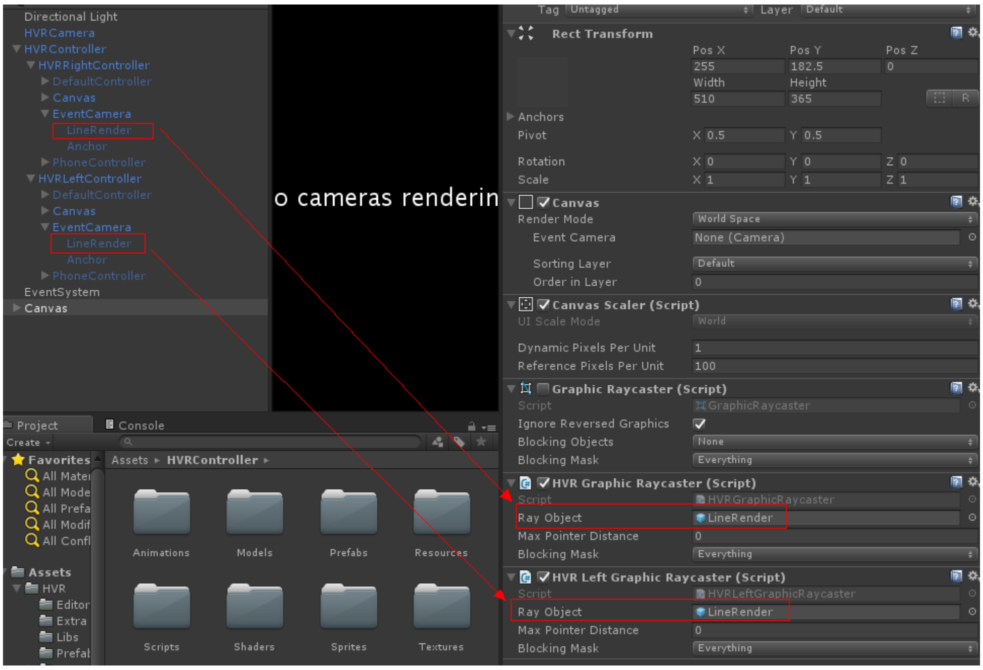


Figure 2-9 Adding the HVR controller prefabs

➢ The HVRController prefabs provide the following functions:

1. Controller ray collision detection:
   - Objects with colliders can respond to collision detection by default.
   - If no collider, such as **BoxCollider**, is added to the UI, deselect the original **Graphic Raycaster** script on the Canvas where the UI is located, bind the **HVRGraphicRaycaster.cs** script, and drag and bind **LineRender** to **RayObject** to respond to collision detection. For dual-controller interaction, bind the **HVRLeftGraphicRaycaster.cs** script, as shown in Figure 2-10.



Tag Untagged Layer Default

Rect Transform

Pos X 255 Pos Y 182.5 Pos Z 0

Width 510 Height 365

Anchors

Pivot X 0.5 Y 0.5

Rotation X 0 Y 0 Z 0

Scale X 1 Y 1 Z 1

☑ Canvas

Render Mode World Space

Event Camera None (Camera)

Sorting Layer Default

Order in Layer 0

☑ Canvas Scaler (Script)

UI Scale Mode World

Dynamic Pixels Per Unit 1

Reference Pixels Per Unit 100

□ Graphic Raycaster (Script)

Script GraphicRaycaster

Ignore Reversed Graphics ☑

Blocking Objects None

Blocking Mask Everything

☑ HVR Graphic Raycaster (Script)

Script HVRGraphicRaycaster

Ray Object LineRender

Max Pointer Distance 0

Blocking Mask Everything

☑ HVR Left Graphic Raycaster (Script)

Script HVRLeftGraphicRaycaster

Ray Object LineRender

Max Pointer Distance 0

Blocking Mask Everything

Figure 2-10 Binding the **Graphic Raycaster** script

- Developers can monitor the collision detection function by referring to section 4.10 in the API description document.

2. For details about how to customize the tooltip displayed when a user presses a button, see section 4.11 in the API description document.
3. For details about simulating 6DOF functions on a 3DOF controller, such as initializing the controller initial position and distance between elbows, see section 4.12 in the API description document.

➢ Special buttons on the controller:

1. For details about how to obtain the long press and short press events, see the implementations in **Assets\HVR\Scripts\HVRControllerUtils.cs**.
2. Controller default buttons and shortcuts

Action	Button	Duration	Function
Press	Home button	< 1s	Access the VRLauncher screen.
Press and hold	Home button	≥ 1s	Reset viewpoint.
Press and hold	Back button	≥ 3s	Access the VRLauncher screen.
Press simultaneously	Home button and trigger button	< 1s	Capture screenshot.

NOTE

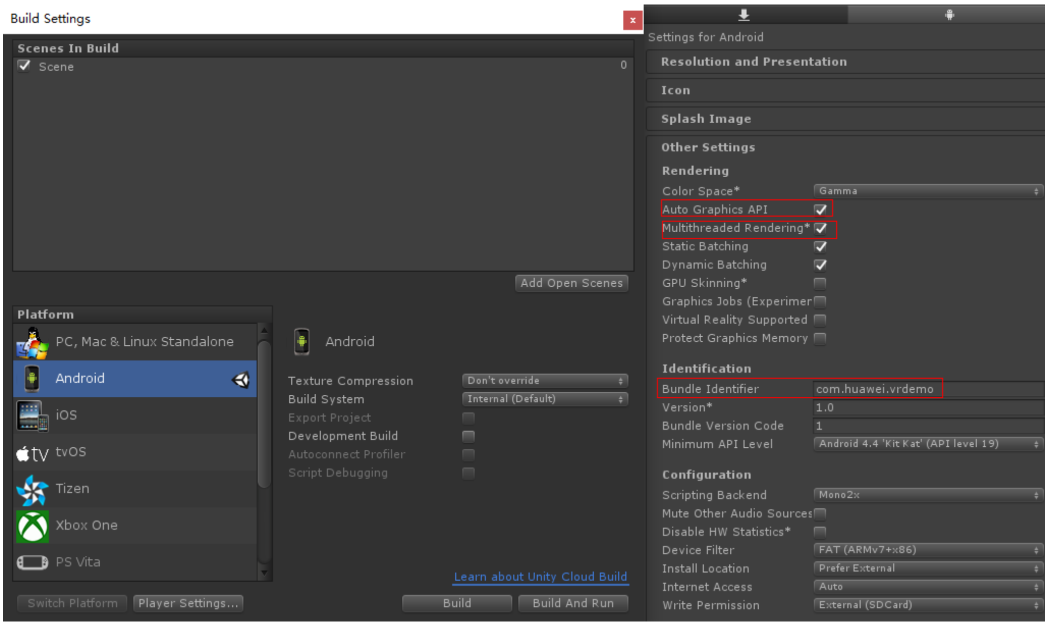
- If you want to customize shortcuts, make sure to avoid the above shortcuts.
- The controller status is updated in the **Update()** method of the **HVRLayoutCore.cs** script. To ensure that all scripts for obtaining the controller status in the app can obtain consistent controller status in every frame, manually adjust the script execution sequence to ensure that those scripts are all executed before or after the SDK script.

3. To manually set the script execution sequence, go to **Edit** > **Project Settings** > **Script Execution Order**, and set numbers for scripts.

By now, the SDK prefabs have all been added.

2.2.4 Compiling and Generating APK

- Go to **File** > **Build Settings...**, and in the **Build Settings** dialog box that is displayed, select **Android** in **Platform**.
- Click **Player Settings...** in the dialog box. On the **Inspector** screen, click **Other Settings** and select **Multithreaded Rendering**.
- The SDK 3.0 does not support **Vulkan**. It is advised that you select **Auto Graphics API**.
- Set the package name in the **Bundle Identifier** field, for example, **com.huawei.vrdemo**, as shown in Figure 2-11.



☑ Scene 0

Texture Compression: Don't override
Build System: Internal (Default)
Export Project □
Development Build □
Autoconnect Profiler □
Script Debugging □

Color Space*: Gamma
Auto Graphics API ☑
Multithreaded Rendering* ☑
Static Batching ☑
Dynamic Batching ☑
GPU Skinning* □
Graphics Jobs (Experimer □
Virtual Reality Supported □
Protect Graphics Memory □

Bundle Identifier: com.huawei.vrdemo
Version*: 1.0
Bundle Version Code: 1
Minimum API Level: Android 4.4 'Kit Kat' (API level 19)

Scripting Backend: Mono2x
Mute Other Audio Sources □
Disable HW Statistics* □
Device Filter: FAT (ARMv7+x86)
Install Location: Prefer External
Internet Access: Auto
Write Permission: External (SDCard)

Figure 2-11 Build settings

Click **Build** in the Build Settings dialog box to generate the APK. Alternatively, you can connect the smartphone to the PC, and click **Build And Run** to directly download and run the APK on the phone.

2.3 Configurations

2.3.1 AndroidManifest Configurations

The **AndroidManifest** file of the app **must** meet the following requirements.

- The following **meta-data** is included in **application** to indicate a Huawei VR app.

```
<meta-data android:name="com.huawei.android.vr.application.mode"

android:value="vr_only"/>
```

- The following **meta-data** is included in **application** to ensure that the content can be properly displayed on the screens with special aspect ratio, such as the Mate 10 Pro.

```
<meta-data android:name="android.max_aspect" android:value="2.1" />
```

- The following **meta-data** is included in **application** to set the app type to "game" or " video". The default type is "game". The SDK adjusts the temperature control mechanism based on the app type.

```
<meta-data android:name="com.huawei.android.vr.application.type" android:value="game" />
```

- The **intent-filter** in **VR Activity** is changed to the following, in order to remove the home screen app icon and enable the redirection in **VrLauncher**.

```
<activity android:name="yourPackage.yourActivity">
  <intent-filter>
    <action android:name="com.huawei.android.vr.action.MAIN"/>
    <category android:name="android.intent.category.DEFAULT"/>
  </intent-filter>
</activity>
```

- The following permission is added for the using of the VR services.

```
<uses-permission android:name="com.huawei.android.permission.VR"/>
```

- The following permission is added for the using of the controller services.

```
<uses-permission android:name="com.huawei.vrhandle.permission.DEVICE_MANAGER" />
```

- The following permission is added for the using of the screenshot function.

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

For details, see the default **AndroidManifest.xml** file in **Assets\Plugins\Android**.

In addition, to view the display on the mobile phone without connecting the glasses, the following **meta-data** must be included in **application** of the **AndroidManifest** file. However, the label **must be removed** from the officially released app. This function is used only for testing and debugging.

```
<meta-data android:name="com.huawei.android.vr.sensor.mode" android:value="mobile"/>
```

2.3.2 VRLauncher Icon Requirements

To display the app icon in VRLauncher, configure as follows:

Solution 1 (recommended): layered effect

Foreground image

- Name: **vr_icon_foreground.png**
- Size: Less than 200 KB
- Format: PNG
- Resolution: 576 x 324
- Directory for storing icons: **Assets/StreamingAssets/vr_icon_foreground.png** of the Unity3D project.

Background image

- Name: **vr_icon_background.png**
- Size: Less than 200 KB
- Format: PNG
- Resolution: 576 x 324
- Directory for storing icons: **Assets/StreamingAssets/vr_icon_background.png** of the Unity 3D project.

Solution 2: Not layered but is compatible with earlier versions, and can coexist with solution 1.

- Name: **vr_icon.png**
- Size: Less than 200 KB
- Format: PNG
- Resolution: 576 x 324
- Directory for storing icons: **Assets/StreamingAssets/vr_icon.png** of the Unity 3D project.

2.3.3 Customizing an Activity

If customized activities are available in compiled Android-based applications, HVRActivity in HUAWEI VR Unity SDK needs to be inherited.

HVRActivity is defined in the **hvrbridge.jar** file in the **Assets\Plugins\Android** directory and is inherited from **UnityPlayerActivity**.

Perform the following operations in an Android project.

```
import com.huawei.vrlab.HVRActivity;

public class MainActivity extends HVRActivity {

……

}
```

In addition, change the activity name to a customized activity name in the **AndroidManifest.xml** file, and ensure that intent-filter meets the requirements in section 2.3.1.

2.3.4 Setting Prompt Information on the 2D Page

SDK3. supports the prompt function on the 2D page. The 2D page is contained in **Assets\Plugins\Android\hvrprompt.aar** and is packaged to the app by default.

This function is required only when a 2D activity is available in an application and a 2D activity needs to be dynamically switched to the VR activity.

You can use the following example codes in your 2D activity to enable the prompt information display. After connecting the helmet, you can start your own VR activity. VRLauncher is skipped, as shown in Figure 2-12.

```
Intent intent = new Intent();

intent.setPackage(getPackageName());

intent.setAction("com.huawei.android.vr.PROMPT");

startActivity(intent);
```

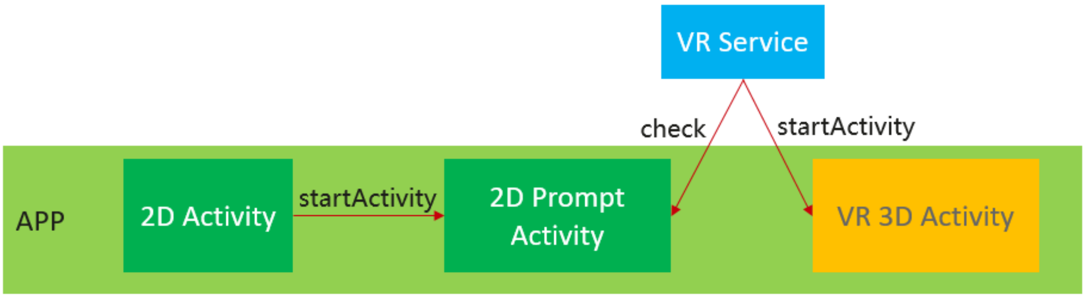


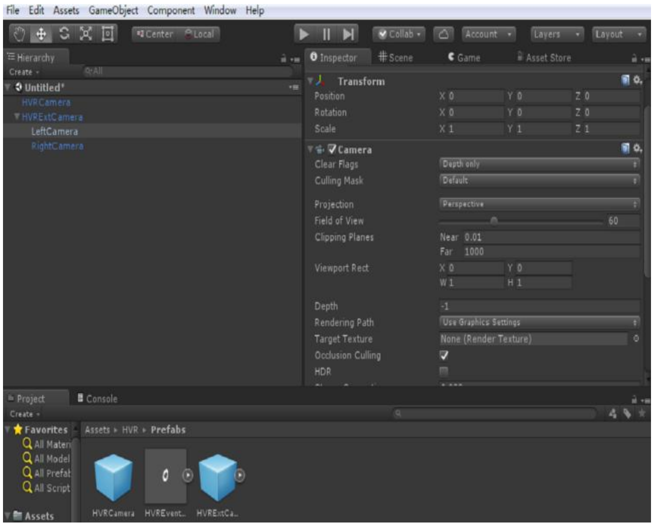
Figure 2-12 Prompt screen redirection

2.4 Functional Options

2.4.1 Integrating Multiple Cameras

If multiple cameras on the Unity need to be integrated, perform the following operations to add the cameras:

Drag **HVRExtCamera** from the **Assets/HVR/Prefabs** directory to the **Hierarchy** area. Under **HVRExtCamera**, two cameras are mounted: **LeftCamera** and **RightCamera**, as shown in Figure 2-13. **LeftCamera** and **RightCamera** correspond to the left and right views of the HUAWEI VR glasses. You can drag multiple groups of cameras named **HVRExtCamera** and modify the parameters of each group of the cameras, such as **Clear Flags**, **Culling Mask**, and **Depth**.



Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

☑ Camera
Clear Flags: Depth only
Culling Mask: Default
Projection: Perspective
Field of View: 60
Clipping Planes: Near 0.01, Far 1000
Viewport Rect: X 0 Y 0, W 1 H 1
Depth: -1
Rendering Path: Use Graphics Settings
Target Texture: None (Render Texture)
Occlusion Culling: ☑
HDR: □

Figure 2-13 Multi-camera prefab setting page

2.4.2 Previewing and Debugging in Unity Editor Play Mode

In Unity Editor Play mode, you can use the keyboard and mouse control to simulate the head movement and controller button status for preview and debugging. The following table lists the corresponding operations.

Head movement

Simulated Movement	Operation
Yaw rotation	Press and hold Alt, and move the mouse leftwards and rightwards.
Pitch rotation	Press and hold Alt, and move the mouse upwards and downwards.
Roll rotation	Press and hold Ctrl, and move the mouse leftwards and rightwards.
Reset yaw rotation	Alt + Z.

Controller buttons

Simulated Movement		Operation
Touch panel Confirm button	Press and hold	Press and hold Shift, and click the mouse left-click button.
	Click	Click the mouse left-click button.
Back button	Press and hold	Press and hold Shift, and click the mouse right-click button.
	Click	Click the mouse right-click button.
Trigger button	Press and hold	Press and hold Shift, and click the mouse wheel.
	Click	Click the mouse wheel.
Swipe up on the touch panel		Press and hold Shift, and click the up arrow key.

Swipe down on the touch panel	Press and hold Shift, and click the down arrow key.
Swipe left on the touch panel	Press and hold Shift, and click the left arrow key.
Swipe right on the touch panel	Press and hold Shift, and click the right arrow key.
Touch on the touch panel	Press and hold Shift.

2.4.3 Power Consumption Reduction Measures

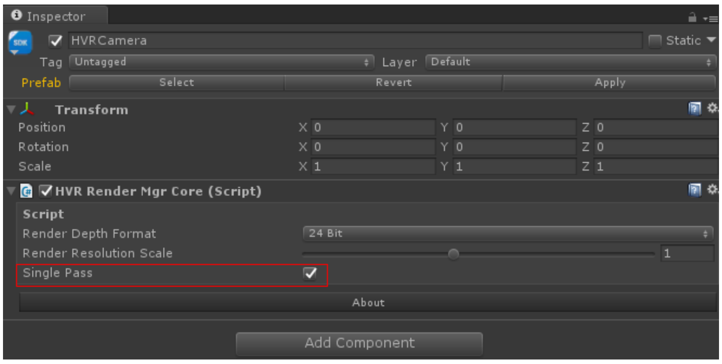
The HUAWEI VR SDK 3.0 provides two power consumption reduction measures, Multiview and monocular rendering, for different VR app scenes. Developers need to select and adapt the two measures with the new APIs.

The reference for the mapping between scenes and APIs are as follows:

- ✧ Videos: monocular rendering.
- ✧ Games: Multiview.
- ✧ Others: Decided based on app types. Use monocular rendering for common tools (such as VR virtual desktop), and Multiview for apps that are highly dependent on 3D effects (such as VR 3D painting).

➢ To enable Multiview mode:

Step 1 Select **SinglePass**, as shown in Figure 2-14.



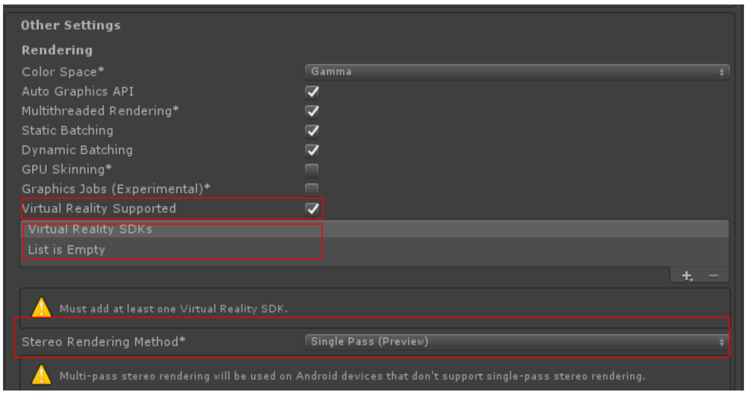
☑ HVRCamera □ Static
Tag Untagged Layer Default

Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

☑ HVR Render Mgr Core (Script)
Script
Render Depth Format 24 Bit
Render Resolution Scale 1
Single Pass ☑

Figure 2-14 Figure 14 Selecting SinglePass

Step 2 Select **Virtual Reality Supported** in **Player Settings**, clear the **VirtualReality SDKs** list, and set **Stereo Rendering Method** to **Single Pass (Preview)**, as shown in Figure 2-15.



Other Settings

Rendering

Color Space*	Gamma
Auto Graphics API	☑
Multithreaded Rendering*	☑
Static Batching	☑
Dynamic Batching	☑
GPU Skinning*	□
Graphics Jobs (Experimental)*	□
Virtual Reality Supported	☑

Virtual Reality SDKs

List is Empty

⚠ Must add at least one Virtual Reality SDK.

Stereo Rendering Method*	Single Pass (Preview)

⚠ Multi-pass stereo rendering will be used on Android devices that don't support single-pass stereo rendering.

Figure 2-15 Setting Stereo Rendering Method

NOTE

When Multiview mode is enabled, the **HVRExtCamera** prefabs cannot be integrated.

- To enable monocular rendering mode:

  Call the **void EnableMonocularDisplay(bool enable)** API. For details about the invoking method, see section 4.8.4 in the API description document.

NOTE

Multiview and monocular rendering modes cannot be enabled at the same time.