Quickstart for Google Cardboard for Unity

This guide shows you how to use the Google Cardboard XR Plugin for Unity (https://github.com/googlevr/cardboard-xr-plugin) for Unity to create your own Virtual Reality (VR) experiences.

Note: Make sure to review the Cardboard branding guidelines (/cardboard/develop/branding-guidelines) before distributing your app to a wider audience.

You can use the Cardboard SDK to turn a mobile phone into a VR platform. A smartphone can display 3D scenes with stereoscopic rendering, track and react to head movements, and interact with apps by detecting when the user presses the viewer button.

To get started, you'll use **HelloCardboard**, a demo game that demonstrates the core features of the Cardboard SDK. In the game, users look around a virtual world to find and collect objects. It shows you how to:

- Set up your development environment
- Download and build the demo app
- Scan the QR code of a Cardboard viewer to save its parameters
- Track the user's head movements
- Render stereoscopic images by setting the correct distortion for each eye
- Turn VR mode on and off

Set up your development environment

Software requirements:

- Unity 2021.3.44f1 (https://unity3d.com/get-unity/download/archive) or later
    - Make sure to include Android and iOS Build Support during installation.
    - Mare sure to install patch version 44f1 or later.
- Git (https://git-scm.com/) must be installed and the `git` executable must be on the `PATH` environment variable. See Unity's package manager git support (https://docs.unity3d.com/Manual/upm-git.html) docs for more details.

Import the SDK and create a new project

Follow these steps to import the Unity SDK and create a new project.

1. Open Unity and create a new **3D** project.
2. In Unity, go to **Window** > **Package Manager**.
3. Click **+** and select **Add package from git URL**.
4. Paste `https://github.com/googlevr/cardboard-xr-plugin.git` into the text entry field. The package should be added to the installed packages.
5. Navigate to the **Google Cardboard XR Plugin for Unity** package. In the **Samples** section, choose **Import into Project**.
   The sample assets should be loaded into `Assets/Samples/Google Cardboard/<version>/Hello Cardboard`.

Note: `<version>` is the `X.Y.Z` semantic version number of the released package (for example, `1.1.0`).

Configuring HelloCardboard scene

1. Navigate to `Assets/Samples/Google Cardboard/<version>/Hello Cardboard/Scenes`, select **Add Open Scenes**, and choose **HelloCardboard** to open the sample scene.
2. Open the **Layers** menu and select **Edit Layers...**.
3. Define a new layer called "Interactive".
4. Click on the **Treasure** GameObject to open the Inspector window. Set its layer to be "Interactive". If a pop up window appears asking if you want to set layer to Interactive for all child objects as well, click on "Yes, change children".
5. Click on the **Player > Camera > CardboardReticlePointer** GameObject to open the Inspector window. In the "Carboard reticle pointer" script, select "Interactive" as the **Reticle Interaction Layer Mask**.

Configuring Android project settings

Navigate to **File** > **Build Settings**.

1. Select **Android** and choose **Switch Platform**.
2. Select **Add Open Scenes** and choose **HelloCardboard**.

Player Settings

Resolution and Presentation

Navigate to **Project Settings** > **Player** > **Resolution and Presentation**.

1. Set the **Default Orientation** to **Landscape Left** or **Landscape Right**.
2. Disable **Optimized Frame Pacing**.

Note: While supported by the Cardboard XR plugin, the **Portrait** and **Portrait Upside Down** orientations may not provide enough room for eye rendering on devices.

Other Settings

Navigate to **Project Settings** > **Player** > **Other Settings**.

1. Choose `OpenGLES2`, `OpenGLES3`, or `Vulkan`, or any combination of them in **Graphics APIs**.
2. Select `Android 8.0 'Oreo' (API level 26)` or higher in **Minimum API Level**.
3. Select `API level 33` or higher in **Target API Level**.
4. Select `IL2CPP` in **Scripting Backend**.
5. Select desired architectures by choosing `ARMv7`, `ARM64`, or both in **Target Architectures**.
6. Select `Require` in **Internet Access**.
7. Select `Input System Package (New)` in **Active Input Handling**.
8. Specify your company domain under **Package Name**.
9. If `Vulkan` was selected as **Graphics API**:
   - Uncheck **Apply display rotation during rendering** checkbox in **Vulkan Settings**.
   - If the Unity version is 2021.2 or above, Select `ETC2` in **Texture compression format**.
10. If the Unity version is 2023.1 or later, select `Activity` and clear `GameActivity` in **Application Entry Point**.

Note: It's possible to use a lower minimum API level by changing rendering API compatibility. For more information, see the SDK's **`build.gradle`**
(https://github.com/googlevr/cardboard/blob/e7a1c22e456ff67116dd016c760dcc469bfa973e/sdk/build.gradle#L11-L18)
.

Note: In case you are experiencing issues when selecting Vulkan as the graphics API, check the `Development Build` box in `Build Settings` and analyze the runtime logs looking for driver compatibility errors.

Note: OpenGL ES 2.0 is no longer supported in Unity 2023.1+. For newer versions use OpenGL ES 3.0 or Vulkan instead. For more information, see the Android requirements and compatibility (https://docs.unity3d.com/2023.3/Documentation/Manual/android-requirements-and-compatibility.html) page in the Unity manual.

Publishing Settings

Navigate to **Project Settings** > **Player** > **Publishing Settings**.

1. In the **Build** section, select `Custom Main Gradle Template` and `Custom Gradle Properties Template`.
2. Add the following lines to the dependencies section of `Assets/Plugins/Android/mainTemplate.gradle`:

```
implementation 'androidx.appcompat:appcompat:1.6.1'
implementation 'com.google.android.gms:play-services-vision:20.1.3'
implementation 'com.google.android.material:material:1.12.0'
implementation 'com.google.protobuf:protobuf-javalite:3.19.4'
```

3. Add the following lines to `Assets/Plugins/Android/gradleTemplate.properties`:

```
android.enableJetifier=true
android.useAndroidX=true
```

Note: The dependencies needed may change between versions. If you want to use a version different from the most recent release, take a look at the history of the dependencies section in sdk/build.gradle (https://github.com/googlevr/cardboard/blob/master/sdk/build.gradle) of the Cardboard SDK repository.

XR Plug-in Management Settings

Navigate to **Project Settings** > **XR Plug-in Management**.

1. Select `Cardboard XR Plugin` under **Plug-in Providers**.

Build your project

Navigate to **File** > **Build Settings**.

1. Select **Build**, or choose a device and select **Build and Run**.

Configuring iOS project settings

Navigate to **File** > **Build Settings**.

1. Select **iOS** and choose **Switch Platform**.
2. Select **Add Open Scenes** and choose **HelloCardboard**.

Player Settings

Resolution and Presentation

Navigate to **Project Settings** > **Player** > **Resolution and Presentation**.

1. Set the **Default Orientation** to **Landscape Left** or **Landscape Right**.

Note: While supported by the Cardboard XR plugin, the **Portrait** and **Portrait Upside Down** orientations may not provide enough room for eye rendering on devices.

Other Settings

Navigate to **Project Settings** > **Player** > **Other Settings**.

1. In **Camera Usage Description**, write `Cardboard SDK requires camera permission to read the QR code (required to get the encoded device parameters).`.
2. In **Target minimum iOS Version**, write `12.0`.
3. Specify your company domain under **Package Name**.

Note: If using an iPhone X, select the **Hide home button on iPhone X** option.

XR Plug-in Management Settings

Navigate to **Project Settings** > **XR Plug-in Management**.

1. Select `Cardboard XR Plugin` under **Plug-in Providers**.

Build your project

Navigate to **File** > **Build Settings**.

1. Select **Build** or **Build and Run**.

Recentering

The Cardboard SDK (https://github.com/googlevr/cardboard) allows you to recenter the head tracker using `Recenter()` (https://github.com/googlevr/cardboard-xr-plugin/blob/master/Runtime/Api.cs).

Follow these steps to try it out using the sample application:

1. Move the device to the position you want to recenter (use as new looking forward head pose).
2. Hold the trigger of your Cardboard device active for at least three seconds.
3. Release the trigger.
4. The initial pose is now in the direction the camera is pointing.

Turning VR mode on and off

The Unity XR Plugin Management API (https://docs.unity3d.com/Manual/com.unity.xr.management.html) lets you turn VR mode on or off for the Google Cardboard XR Plugin for Unity (https://github.com/googlevr/cardboard-xr-plugin). End-user documentation and usage examples are available in Unity's End-user documentation (https://docs.unity3d.com/Packages/com.unity.xr.management@3.2/manual/index.html).

The **VrMode** scene in HelloCardboard sample shows a basic usage of the aformentioned API. In this scene, VR mode can be turned off by tapping **exit**
✕
, and can be turned on again just by tapping anywhere on the screen. Check VrModeController.cs (https://github.com/googlevr/cardboard-xr-plugin/blob/master/Samples%7E/hellocardboard-unity/Scripts/VrModeController.cs)
for details about how this is performed.

Next steps

- Review the Cardboard branding guidelines (/cardboard/develop/branding-guidelines).

Last updated 2024-12-09 UTC.