**NOLO VR UE4 SDK**

**Documentation**

NOLO Co., Ltd

March 2018

CONTENTS

[1. Overview 4](#_Toc510200922)

[1.1 About NOLO 4](#_Toc510200923)

[1.2 About NOLO CV1 4](#_Toc510200924)

[1.3 About NOLO HOME 5](#_Toc510200925)

[1.4 About NOLO VR UE4 SDK 5](#_Toc510200926)

[1.5 NOLO VR UE4 SDK Framework 5](#_Toc510200927)

[2. Set Up Development Environment 7](#_Toc510200928)

[3. API Description 8](#_Toc510200929)

[3.1 Blueprint Description 8](#_Toc510200933)

[3.2 Detailed Description 9](#_Toc510200934)

[3.3 Controller Buttons 12](#_Toc510200935)

[4. NOTES 16](#_Toc510200936)

[4.1 Set Origin 16](#_Toc510200938)

[4.2 Set AppKey 16](#_Toc510200939)

[4.3 Reset Orientation 17](#_Toc510200940)

[4.4 UE4 Camera Settings 17](#_Toc510200941)

[4.5 Working With Third-Party VR SDK 18](#_Toc510200942)

# Overview

## About NOLO

NOLO is dedicated to combine desktop-grade VR gaming experience with the convenience of mobile VR devices, redefining a mobile VR gaming experience like never before.

NOLO kit is compatible with some 87,000,000 VR headsets of all kinds currently on the market, indicating huge market potential. In addition, we’ve partnered with VR headset companies, robotic companies, and drone companies around the globe.

## About NOLO CV1

NOLO CV1 is a short-range high-accuracy motion tracking equipment kit for VR/AR gaming systems, composed of a Base Station, a headset marker and 2 controllers.

NOLO CV1 is compatible with all mainstream mobile VR headsets, some PC VR/AR headsets, and All-In-One’s based on NibiruOS, providing position information and interaction functionality to these devices. With simple setup at first launch, users can move around in the virtual world like in real world, and interact with virtual objects using our hand controllers.

## About NOLO HOME

NOLO HOME is the first ever 6-DoF mobile VR gaming platform, developed by NOLO. With NOLO HOME, NOLO CV1, and a decent smartphone, users can enjoy any VR content (e.g. a VR game on your phone) whenever, wherever, wirelessly, creating a whole new way of entertainment accessible to and affordable for all.

NOLO HOME Android version is now available to the public, iOS version is currently under development.

## About NOLO VR UE4 SDK

NOLO VR UE4 SDK is developed by NOLO Co., Ltd, to provide data like positional information from NOLO devices for UE4 developers. With this SDK, you can develop 6-DoF mobile VR games that run on smartphones or GearVR(GearVR SDK required).

## NOLO VR UE4 SDK Framework

NOLO VR UE4 SDK incorporates a C/S framework. NOLO HOME (server) reads and processes data obtained directly from NOLO CV1 devices. Game (client) establishes a connection with NOLO HOME via AIDL to obtain data from NOLO CV1 (through SDK/NOLO HOME). Therefore, the game app needs NOT read any data directly from USB cable (figure 1).

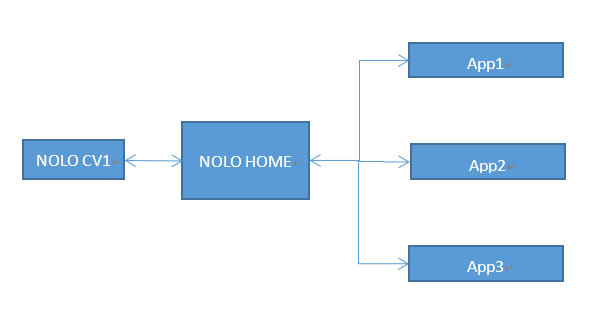


figure 1

NOLO VR Unity SDK and NOLO VR UE4 SDK are built upon NOLO VR Android SDK. NOLO VR Android SDK establishes a communication cannel with NOLO HOME via AIDL, to obtain data from NONO CV1 (figure 2).

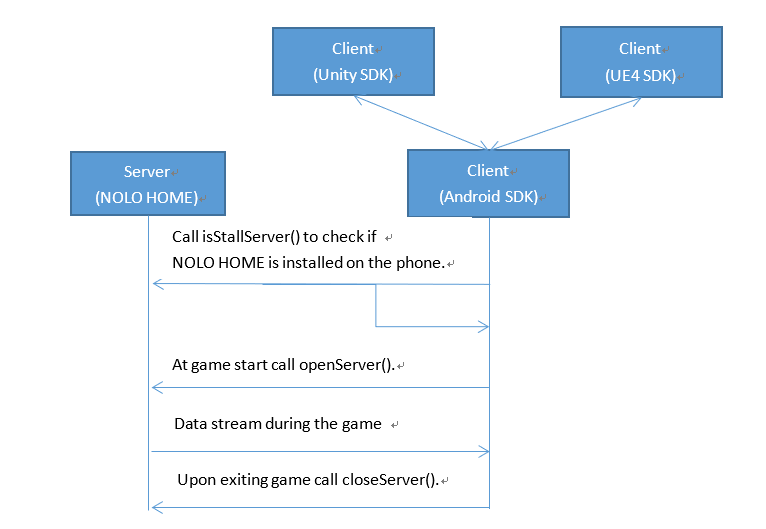


figure 2

# Set Up Development Environment

Requires UnrealEngine4.18.2/4.18.3

Requires NOLO HOME be installed on your test phone(Android)

# API Description



## Blueprint Description

(figure 3)

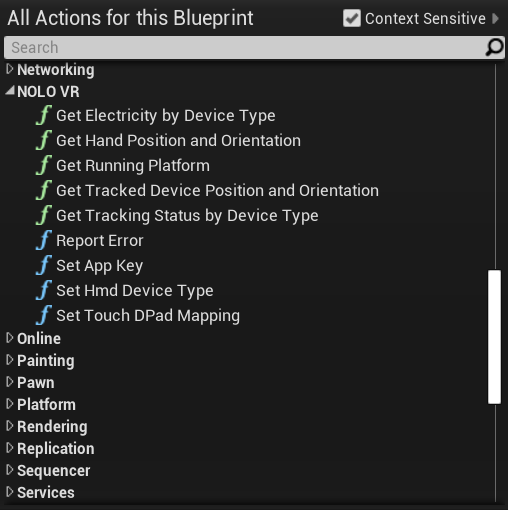


figure 3

1. GetRunningPlatform:Get gaming platform.
2. GetTrackedDevicePositionAndOrientation:To acquire headset marker’s position and orientation.
3. GetHandPositionAndOrientation: To acquire hand controller’s position and orientation.
4. GetTrackingStatusByDeviceType: Get device tracking information (position, rotation, etc.)
5. GetElectricityByDeviceType:To check battery status of a designated device.
6. SetHmdDeviceType: Set headset type.
7. SetAppKey: Check Appkey and decide whether the app is authorized to get NOLO devices’ tracking data.
8. ReportError: Log error messages
9. SetTouchDPadMapping: Define functionality of the 4 keys on Touchpad (top, bottom, left, right).

## Detailed Description

|  |  |
| --- | --- |
| **function name** | **FString GetRunningPlatform（）** |
| **function description** | Get gaming platform. |
| **input parameters** | **NULL** |
| **return value** | **FString : Android or Default** |
| **prerequisites** |  |

|  |  |
| --- | --- |
| **function name** | **bool GetTrackedDevicePositionAndOrientation（FVector & OutPosition, FRotator & OutOrientation）** |
| **function description** | To acquire headset marker’s position and orientation. |
| **input parameters** | **OutPosition : (out) Current position of the device**  **OutOrientation (out) Current orientation of the device** |
| **return value** | **Bool : True if the specified device id had a valid tracking pose this frame, false otherwise** |
| **prerequisites** |  |

|  |  |
| --- | --- |
| **function name** | **Bool GetHandPositionAndOrientation（EControllerHand Hand, FVector& OutPosition, FRotator& OutOrientation）** |
| **function description** | To acquire hand controller’s position and orientation. |
| **input parameters** | **Hand : Which hand's controller to get the position and orientation for**  **OutPosition : (out) Current position of the device**  **OutOrientation (out) Current orientation of the device** |
| **return value** | **Bool : True if the specified controller is Available** |
| **prerequisites** |  |

|  |  |
| --- | --- |
| **function name** | **bool GetTrackingStatusByDeviceType(const ENoloDeviceType DeviceType)** |
| **function description** | Get device tracking information (position, rotation, etc.) |
| **input parameters** | **ENoloDeviceType : Hmd,Left,Right,Basestation** |
| **return value** | **True : Tracked ,false : not Tracked** |
| **prerequisites** |  |

|  |  |
| --- | --- |
| **function name** | **Bool bool GetElectricityByDeviceType(const ENoloDeviceType DeviceType,int & battery)** |
| **function description** | To check battery status of a designated device. |
| **input parameters** | **ENoloDeviceType : Hmd,Left,Right,Basestation**  **Battery : (out)0:shut down 、1:low 、2:middle 、3:high** |
| **return value** | **Bool : True if the battery info is Available** |
| **prerequisites** |  |

|  |  |
| --- | --- |
| **function name** | **void SetHmdDeviceType(const EMobileVR\_HMDDeviceType DeviceType)** |
| **function description** | Set headset type. |
| **input parameters** | **EMobileVR\_HMDDeviceType : DT\_GearVR、 DT\_GoogleVR\_Cardboard、DT\_GoogleVR\_Daydream、DT\_Other** |
| **return value** | **Void** |
| **prerequisites** |  |

|  |  |
| --- | --- |
| **function name** | **void SetAppKey (FString appKey)** |
| **function description** | Check Appkey and decide whether the app is authorized to get NOLO devices’ tracking data. |
| **input parameters** | **FString** |
| **return value** | **Void** |
| **prerequisites** |  |

|  |  |
| --- | --- |
| **function name** | **void ReportError (FString appKey)** |
| **function description** | Log error messages |
| **input parameters** | **FString** |
| **return value** | **Void** |
| **prerequisites** |  |

|  |  |
| --- | --- |
| **function name** | **void SetTouchDPadMapping (ENoloVRTouchDPadMapping NewMapping)** |
| **function description** | Define functionality of the 4 keys on Touchpad (top, bottom, left, right). |
| **input parameters** | **ENoloVRTouchDPadMapping** |
| **return value** | **Void** |
| **prerequisites** |  |

## Controller Buttons

(figure 4) (figure 5) (figure 6).

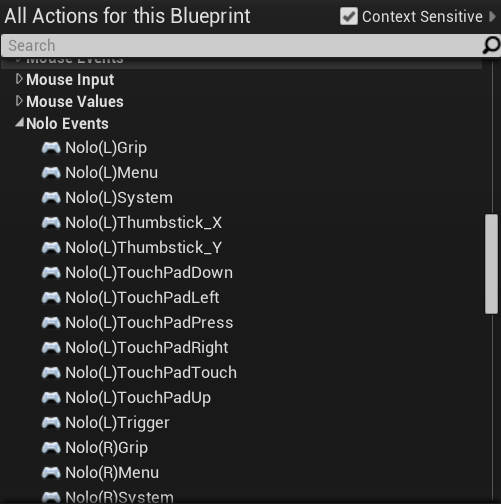


figure 4

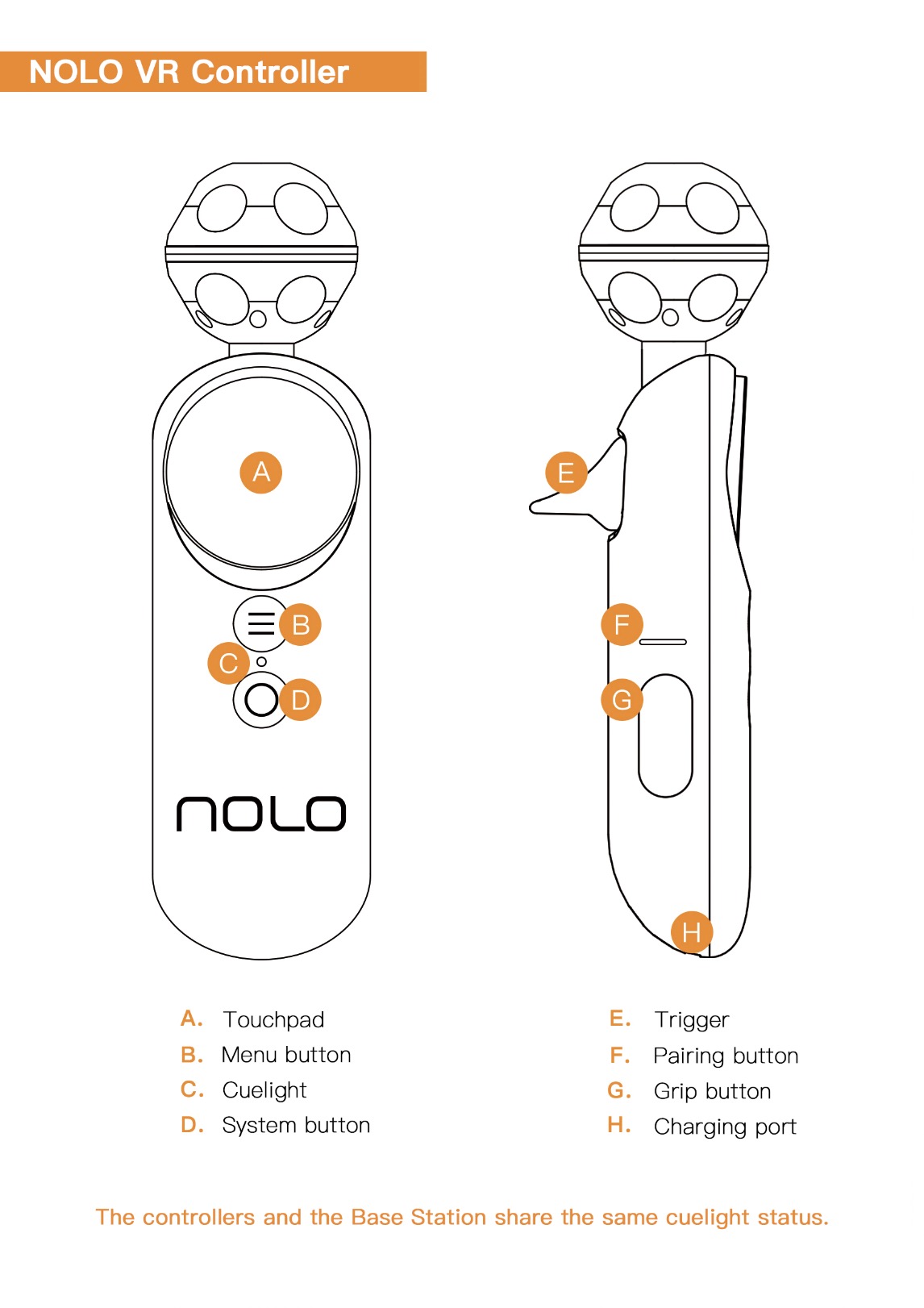


figure 5

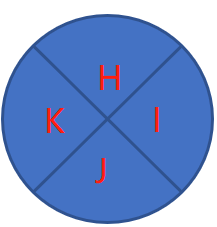
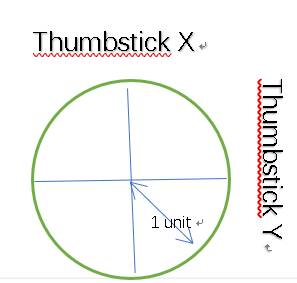


figure 6

Button Table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Mark** | **Description** | **NOLO Events** | **Gamepad Events** | **SteamEvents** |
| A | TouchPadPress | Nolo\_Left\_TouchPadPress | MotionController\_Left\_Thumbstick |  |
| A | TouchPadTouch | Nolo\_Left\_TouchPadTouch |  | Steam\_Touch\_0 |
| B | Menu | Nolo\_Left\_Menu | MotionController\_Left\_Shoulder |  |
| D | System | Nolo\_Left\_System | SpecialLeft |  |
| E: | Trigger | Nolo\_Left\_Trigger | MotionController\_Left\_Trigger |  |
| G | Grip | Nolo\_Left\_Grip | MotionController\_Left\_Grip1 |  |

Touchpad Table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Mark** | **Description** | **NOLO Events** | **Face Buttos**  **(Gamepad Events)** | **ThumbstickDirections**  **(Gamepad Events)** |
| H | Up | Nolo\_Left\_TouchPadUp | MotionController\_Left\_FaceButton1 | MotionController\_Left\_Thumbstick\_Up |
| I | Right | Nolo\_Left\_TouchPadRight | MotionController\_Left\_FaceButton2 | MotionController\_Left\_Thumbstick\_Right |
| J | Down | Nolo\_Left\_TouchPadDown | MotionController\_Left\_FaceButton3 | MotionController\_Left\_Thumbstick\_Down |
| K | Left | Nolo\_Left\_TouchPadLeft | MotionController\_Left\_FaceButton4 | MotionController\_Left\_Thumbstick\_Left |

# NOTES



## Set Origin

Turn on all NOLO devices, place the headset marker on the ground, press the button on the headset marker. The headset marker’s current position will be the origin in the game. The origin’s coordinates will be saved. This process only needs to be repeated if the Base Station has been moved.

## Set AppKey

A game must acquire an AppKey to run properly with NOLO CV1. An AppKey will be generated automatically when developers apply for their game on NOLO Developer Center.

When the game does not upload NOLO HOME, you can use this public Appkey for development testing.

Public Appkey：4e4f4c4f484f4d457eff82725bc694a5

## Reset Orientation

Upon starting a game, if the forward direction in the game does not point towards Nolo Base Station, or the controller orientation seems a little odd, you may need to reset orientation by doing the following: Put on your headset, face the Nolo Base Station, point both controllers towards the Nolo Base Station, then double click the power button on either controller.

## UE4 Camera Settings

NOLO VR UE4 SDK provides 6DoF position/motion data from NOLO CV1. However, it does not pair with UE4 HMD (IHeadMountedDisplay). There are 2 solutions:  
 1. If you choose to use only the positional data (instead of both positional and rotational data) from NOLO’s headset marker, you may design the event graph as follows, aka alter the positional data of the camera’s parent node. In the end, it uses the smartphone’s IMU data for camera rotation, and NOLO headset marker’s positional data for camera position.

2. If you choose to use both positional and rotational data from NOLO’s headset marker, please uncheck “Lock to Hmd”, and use GetTrackedDevicePositionAndOrientation blueprint API we provided to control camera’s position and rotation.

We recommend using solution 1.( (figure 7))

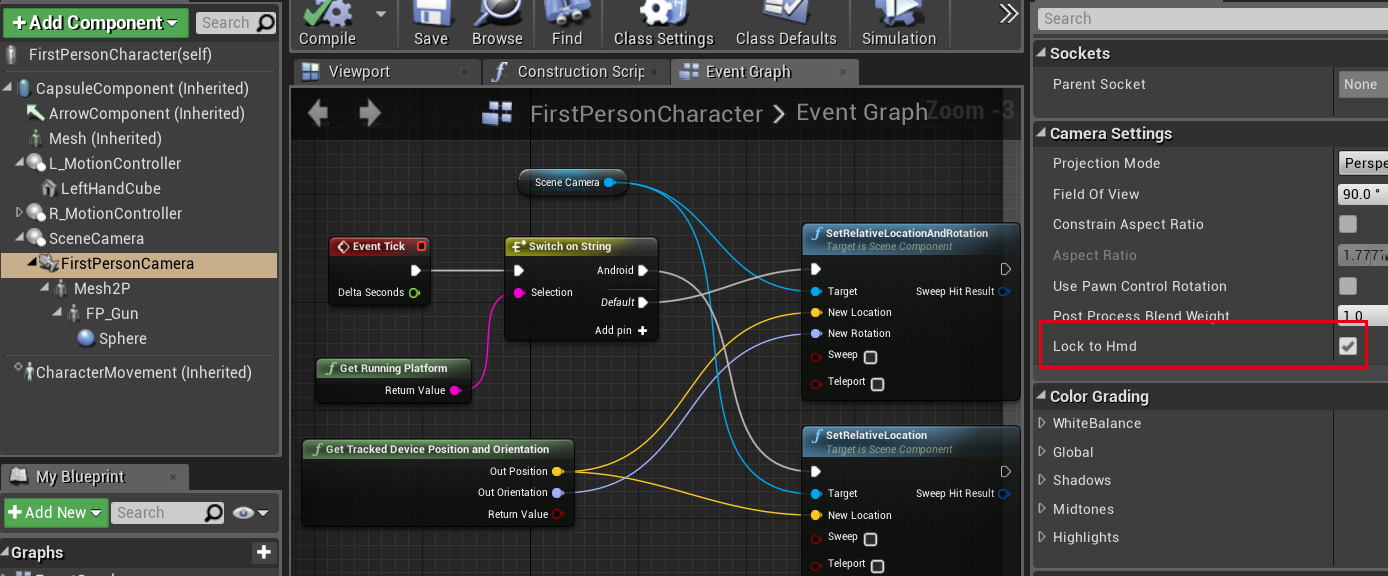


figure 7

## Working With Third-Party VR SDK

NOLO VR UE4 SDK pairs with all UE4’ controller APIs, and provides a blueprint for headset marker. It works with Google VR, Gear VR and gives a 6DoF solution for smartphones (which only have 3DoF data).