**NOLO VR Unity SDK**

**インターフェースについて**

NOLO Co., Ltd

March 2018

目次

[1. 紹介 3](#_Toc510201287)

[1.1 NOLOについて 3](#_Toc510201288)

[1.2 NOLO CV1について 3](#_Toc510201289)

[1.3 NOLO HOMEについて 4](#_Toc510201290)

[1.4 NOLO VR Unity SDKについて 4](#_Toc510201291)

[1.5 モバイル端末のSDK構造 4](#_Toc510201292)

[2. 接続準備 6](#_Toc510201293)

[3. インターフェースについて 7](#_Toc510201294)

[3.1 ボタンイベントログ 7](#_Toc510201295)

[3.2 Touchイベントログ 7](#_Toc510201296)

[3.3 振動イベントログ 8](#_Toc510201297)

[3.4 位置データ 9](#_Toc510201298)

[3.5 エラー情報を提出する。 9](#_Toc510201299)

[4. 注意事項 9](#_Toc510201300)

[4.1 原点位置を設定する。 9](#_Toc510201301)

[4.2 設置 AppKey 10](#_Toc510201302)

[4.3 配置 AndroidManifest.xml 10](#_Toc510201303)

[4.4 cameraパラメーターの修正 10](#_Toc510201304)

[4.5 方向設定 11](#_Toc510201305)

[4.6 ワンクリック 11](#_Toc510201306)

# 紹介

## NOLOについて

NOLOはVRボードゲームの素晴らしい体験及びVR移動の利便性を結びつけ、次世代のVR移動システムのインタラクティブな操作方法を改めて定義する。

現在、NOLOは世界で計8,700万個のVRヘルメットに対応でき、非常に市場の将来性がある。それから、NOLOは国内外で数多くの一体型VRヘルメット、PC VR、AR、サービスロボット及びドローンの関連会社と連携している。

## NOLO CV1について

NOLO CV1はNOLO Inc.自社で開発された世界初の完全没入型VR/AR 製品で、BS1つ、ヘルメットロケーター1つ及びコントローラー2つを含む。

NOLO CV1は現在市販されるVRヘルメット、一部のPC VR、ARのヘルメット及びNibiruOSの一体型に対応でき、空間定位及びインタラクティブの機能を支える。ユーザーは初めて使用する場合、簡単に設置すれば、仮想の現実世界を体験し、コントローラーで仮想世界との多様なインタラクションを体験することができる。

## NOLO HOMEについて

NOLO HOME は NOLO傘下のモバイルVRプラットフォームであり、世界初の携帯電話用6-DoFモバイルVRプラットフォームである。ユーザーはNOLO HOME 、NOLO CV1 及び携帯電話を利用して、時間や場所を問わず、無線で完全な没入型VR体験を享受することができる。一般消費者のために斬新なゲームプレー方法を提供する。現在、Androidのユーザーはダウンロードできる。iOS 版もまもなく使用できる。

## NOLO VR Unity SDKについて

NOLO VR Unity SDKはUnityの開発者がNOLO設備ﾃﾞｰﾀを取得し、Cardboard及びGearVR のSDKを搭載し、モバイル6DoF VRゲームを開発するために、NOLO Inc.により開発されたものである。

## モバイル端末のSDK構造

NOLO SDKはC/S構造を採用する。NOLO HOMEはServeであり、NOLO CV1設備ﾃﾞｰﾀを読込み、処理する。ゲームはClientである。SDKはAIDLによってゲームとNOLOHOMEの間にﾃﾞｰﾀを転送させ、NOLO CV1設備ﾃﾞｰﾀを取得する。ゲームはUSB設備ﾃﾞｰﾀを読み込む必要がなく、NOLO HOMEとの接続のみによってﾃﾞｰﾀの送受信をする。(図 1)

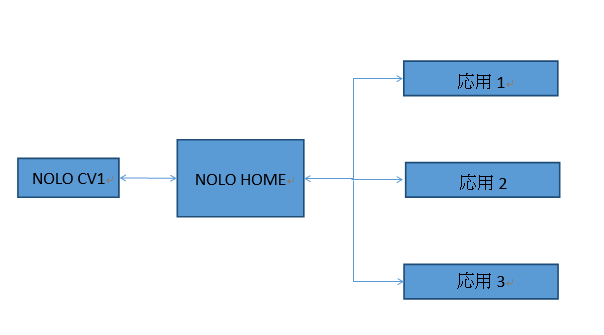


図 1

NOLO VR Android SDKはAIDL及びNOLO HOMEでプロセス間通信を実現することによって、設備ﾃﾞｰﾀを取得する。getInstance()でNOLOVRの実例を取得する。ゲーム開始前にisStallServer()でユーザーがNOLO HOMEをインストールしたかどうかを判断する。インストールを確認された場合、openServer()でServerをバインディングする。ゲーム終了後、closeServer()でバインディングを外す。NOLO VR Unity SDK及びNOLO VR UE4 SDKはNOLO VR Android SDKに対して、部分のインタフェースに対してカプセル化する。開発者はインターフェースを通じて、位置及びボタン値を取得すれば、ゲームを開発することができる。 (図2).

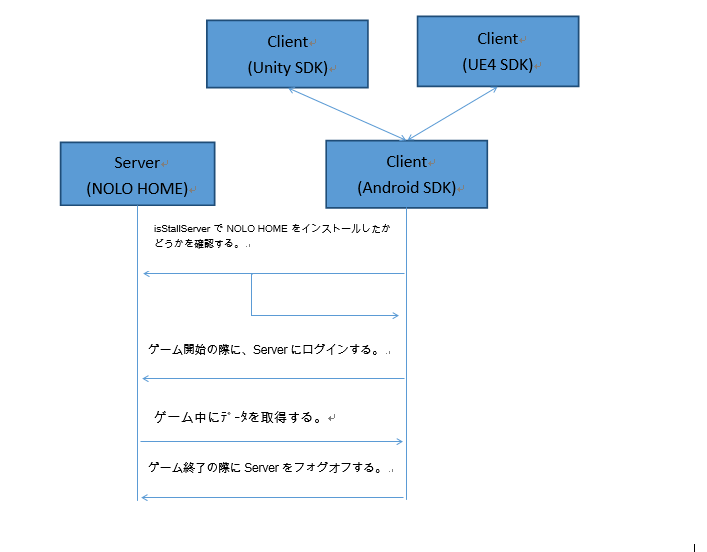


図 2

# 接続準備

開発者を準備する必要Unity5.6以上のバージョンは、NOLO VR Unity SDK。テストの場合、NOLO Driver For Windowsインストールが必要。携帯電話で実行する場合、NOLO HOMEアプリケーションをインストールする必要がある。

# インターフェースについて

## ボタンイベントログ

|  |  |
| --- | --- |
| **関数名** | bool GetNoloButtonPressed() |
| **機能説明** | ボタン長押し |
| **パラメーター** | Enum NoloButtonID |
| **戻り値** | bool |
| **先決条件** | NoloVR\_Controller.GetDevice() |

|  |  |
| --- | --- |
| **関数名** | bool GetNoloButtonDown() |
| **機能説明** | ボタンの押しから外しまでの状態を取得する。 |
| **パラメーター** | Enum NoloButtonID |
| **戻り値** | bool |
| **先決条件** | NoloVR\_Controller.GetDevice() |

|  |  |
| --- | --- |
| **関数名** | bool GetNoloButtonUp() |
| **機能説明** | ボタンの外しから押しまでの状態を取得する。 |
| **パラメーター** | Enum NoloButtonID |
| **戻り値** | bool |
| **先決条件** | NoloVR\_Controller.GetDevice() |

## Touchイベントログ

|  |  |
| --- | --- |
| **関数名** | bool GetNoloTouchPressed() |
| **機能説明** | タッチパネルのタッチ状態を取得する |
| **パラメーター** | Enum NoloTouchID |
| **戻り値** | bool |
| **先決条件** | NoloVR\_Controller.GetDevice() |

|  |  |
| --- | --- |
| **関数名** | bool GetNoloTouchDown() |
| **機能説明** | タッチパネルのタッチなしをタッチ有りにする。 |
| **パラメーター** | Enum NoloTouchID |
| **戻り値** | bool |
| **先決条件** | NoloVR\_Controller.GetDevice() |

|  |  |
| --- | --- |
| **関数名** | bool GetNoloTouchUp() |
| **機能説明** | タッチパネルのタッチ有りをタッチ無しにする |
| **パラメーター** | Enum NoloTouchID |
| **戻り値** | bool |
| **先決条件** | NoloVR\_Controller.GetDevice() |

|  |  |
| --- | --- |
| **関数名** | Vector2 GetAxis() |
| **機能説明** | タッチパネルのタッチ座標を取得する。 |
| **パラメーター** | Enum NoloTouchID: デフォルト値はタッチパネルのパラメーターとする。その他パラメーターは無効となる。（添付資料参照） |
| **戻り値** | Vector2 |
| **先決条件** | NoloVR\_Controller.GetDevice() |

## 振動イベントログ

|  |  |
| --- | --- |
| **関数名** | void TriggerHapticPulse() |
| **機能説明** | コントローラーの振動をコールする。 |
| **パラメーター** | int:振動強度、範囲0~100 |
| **戻り値** | void |
| **先決条件** | NoloVR\_Controller.GetDevice() |

## 位置データ

|  |  |
| --- | --- |
| **関数名** | Nolo\_Transform GetPose() |
| **機能説明** | 設備位置情報を取得する |
| **パラメーター** | Null |
| **戻り値** | Nolo\_Transform |
| **先決条件** | NoloVR\_Controller.GetDevice() |

## エラー情報を提出する。

|  |  |
| --- | --- |
| **関数名** | void ReportError () |
| **機能説明** | LOGｹﾞｰﾑの中のエラー情報を収集する。 |
| **パラメーター** | string |
| **戻り値** | void |
| **先決条件** | NoloVR\_Playform.GetInstance() |

# 注意事項

## 原点位置を設定する。

NOLO正常に稼動する場合、ヘルメットを床に置き、ロケーターのボタンを押せば、当位置がｹﾞｰﾑの初期位置となる。即ち、ゲームエンジン“”の位置。原点位置を設定すれば、設備に登録される。BSの位置が大きく変更した場合のみ、改めて原点位置を設定する必要がある。

## 設置 AppKey

セッティングappkeyの後、ｹﾞｰﾑプレイができる。appkeyそれは開発者がプラットフォームを利用する際に自動生成される。

## 配置 AndroidManifest.xml

おAndroidManifest.xml中のスペックは権限:

<uses-permission android:name="android.permission.BROADCAST\_STICKY" />

<uses-permission android:name="nolo.permission.ACCESS\_SERVER" />

<uses-permission android:name="android.permission.PACKAGE\_USAGE\_STATS" />

<uses-permission android:name="android.permission.SYSTEM\_ALERT\_WINDOW" />

<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />

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

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

<uses-permission android:name="com.android.launcher.permission.WRITE\_SETTINGS" />

<uses-permission android:name="android.permission.WRITE\_APN\_SETTINGS" />

## cameraパラメーターの修正

(図3)

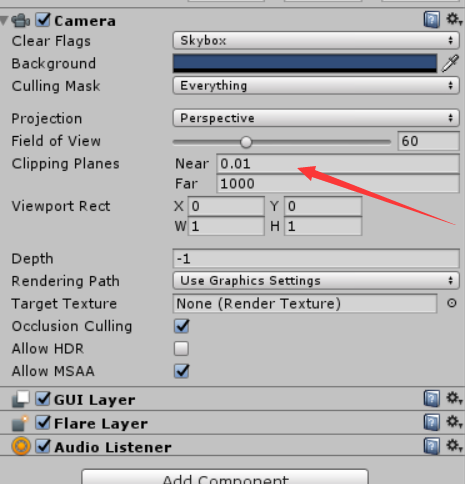


図 3

## 方向設定

ゲーム開始後、ゲームの方向がBSの方向と異なる場合、BSに向かってコントローラーの電源をダブルクリックし、方向を設定する。コントローラーの方向が異なる場合、BSに向かって当コントローラーの電源ｽｲｯﾁをダブルクリックし、方向を設定する。

## ワンクリック

数値を修正すれば、あるボタンを180度回転可能のホットキーにすることができる。当ボタンをダブルクリックする場合、その場で180度回転の操作ができる。ゲームの後ろの背景は目の前となる。( 図 4).



図 4