

# Project Work Intelligent Systems: Firenze su Oculus

PAOLO MARINO & EMANUELE CASCIARO

June 2023

## Contents

<b>1</b>	<b>Introduzione</b>	<b>1</b>
<b>2</b>	<b>Correzione dei modelli</b>	<b>2</b>
2.1	Possibili soluzioni . . . . .	2
<b>3</b>	<b>Unreal Engine Seutp</b>	<b>2</b>
<b>4</b>	<b>Unreal Import</b>	<b>6</b>
4.1	Struttura dei modelli . . . . .	6
<b>5</b>	<b>Terreno</b>	<b>8</b>
5.1	Texture . . . . .	8
5.2	Modello del terreno da scansione altimetrica . . . . .	8
<b>6</b>	<b>Oculus</b>	<b>10</b>
6.1	Set-up . . . . .	10
6.2	Controls . . . . .	11
6.2.1	Game Menu . . . . .	11
6.2.2	Character movement . . . . .	14
<b>7</b>	<b>Facade texturization</b>	<b>17</b>
<b>8</b>	<b>Compilazione e Installazione su Oculus</b>	<b>20</b>
8.1	Abilitare il supporto alla compilazione . . . . .	20
8.2	Abilitare il debug usb nell'oculus . . . . .	22
8.3	Installazione dell'APK . . . . .	22

## 1 Introduzione

L'obiettivo finale dell'elaborato è ottenere un modello di una parte di firenze da poter esplorare con un visore di realtà aumentata. Per fare ciò occorre fornire

modelli 3D adeguati e a partire da questi andare a costruire un semplice modello di "gioco" sul motore grafico di Unreal Engine (eseguito con la versione UE 5.2).

## 2 Correzione dei modelli

Prima di poter procedere all'inclusione dei modelli è necessario correggerne alcuni, dato che durante il processo di export tra il modello SketchUp (programma con il quale sono stati realizzati i modelli che ci sono stata forniti) in formato ".glb" (compatibile con UE sia dal punto di vista delle texture sia dal punto di vista geometrico) avvengono degli errori di texturizzazione, generalmente derivati da un incorretto posizionamento della texture su SketchUp.

### 2.1 Possibili soluzioni

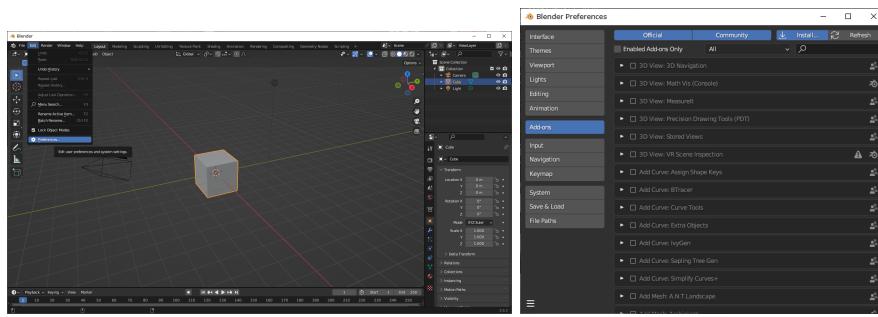
Per risolvere tale problema ci sono differenti soluzioni, ciascuna da usare in caso la precedente non funzioni:

1. selezionare tutto il modello da SketchUp e applicare il plugin di correzione delle facce (installabile su SketchUp, reperibile nella cartella **plugins/FixReversedFaceMaterials\_v1.8.rbz**) →importare il modello di SketchUp su Blender →esportare il modello in formato GLB.
2. selezionare ESCLUSIVAMENTE le facce interessate nel modello da SketchUp e applicare il plugin di correzione delle facce →importare il modello di SketchUp su Blender →esportare il modello in formato GLB.
3. esportare da SketchUp il modello in formato Wavefront (.obj), importare tale formato su Blender e da lì esportarlo come GLB.
4. selezionare ESCLUSIVAMENTE E SINGOLARMENTE (il plugin non è perfetto, può capitare che delle facce lo facciano crashare, in caso occorre applicare usando il pannello sulla destra la texture su entrambi i lati) le facce interessate nel modello da SketchUp e applicare il plugin di correzione delle facce →importare il modello di SketchUp su Blender →esportare il modello in formato GLB.

## 3 Unreal Engine Seutp

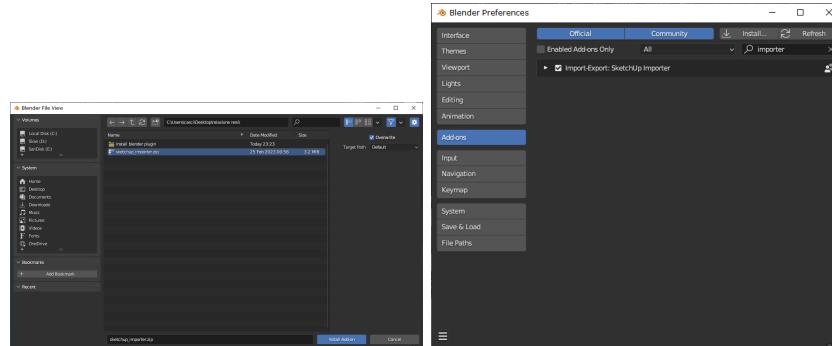
Il lavoro è stato svolto utilizzando Unreal Engine versione 5.2, con API di realtà aumentata di OpenXR: il setup segue la procedura riportata in figura 2. Il motore è incluso con il launcher di EpicGames, con installer allegato nel file **epic\_games\_installer.msi**.

Una volta installato il motore ed il plugin, occorre attivare quest'ultimo. Per fare ciò occorre aprire un qualsiasi progetto dalla schermata principale del launcher, e seguire i passaggi riportati in figura 3



(a) Dall'editor aprire il menu Modifica  
→ Preferenze

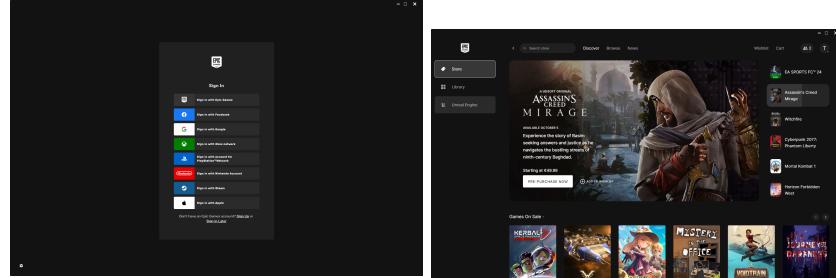
(b) Nell menu Add-Ons, selezionare Installa



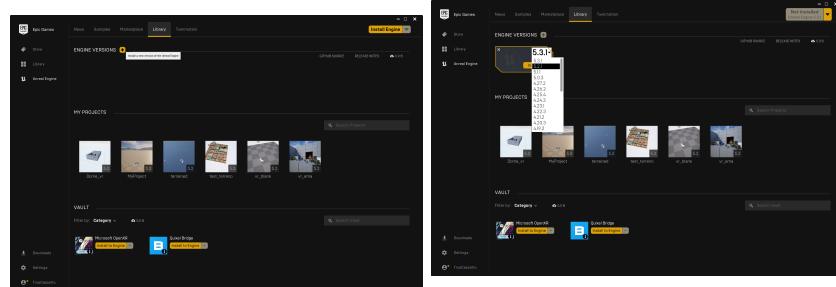
(c) Selezionare il file **plug-ins/sketchup\_importer.zip**  
selezionare Installa

(d) Cercare il plugin appena installato con la barra di ricerca e abilitarlo tramite checkbox

Figure 1: Processo di installazione del plugin di blender per importare modelli SketchUp.

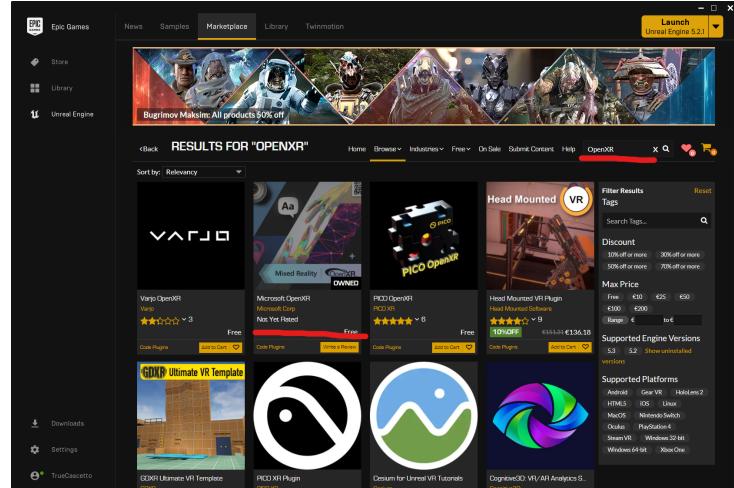


(a) Al primo avvio del launcher, procedere al login con un account EpicGames. L'account non vincola in alcun modo i progetti creati, è possibile modificarli con account differenti.



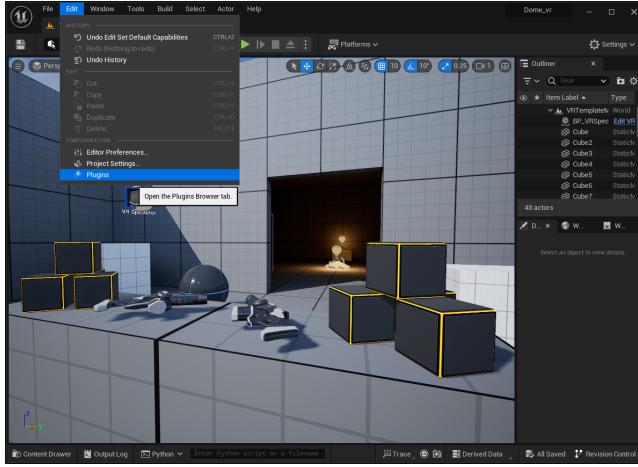
(c) Andare ad installare il motore di UE, selezionando l'icona "+".

(b) Dopo aver effettuato l'accesso, è possibile entrare nel launcher di UE selezionandolo dalla barra laterale.

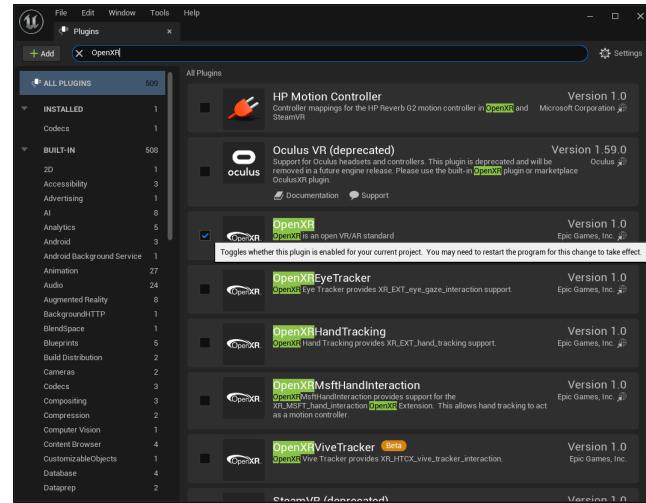


(e) Dal menu MARKETPLACE , usare la barra di ricerca per installare il plugin OpenXR.

Figure 2: Installazione di UE5.2



(a) Dalla barra degli strumenti selezionare EDIT → PLUGINS.



(b) Cercare usando la barra di ricerca il plugin OpenXR ed assicurarsi che la casella associata sia spuntata.

Figure 3: Abilitazione del plugin OpenXR

## 4 Unreal Import

Una volta ottenuti dei modelli utilizzabili per la creazione del mondo è necessario importarli all'interno dell'UE editor. In caso di un unico file GLB contenente tutta la città sarebbe possibile importarlo tramite FILE → IMPORT INTO LEVEL, però è una soluzione da evitare, in quanto UE gestisce le collisioni con la bounding box minimale che copre l'intera mesh, di conseguenza diventa impossibile muoversi all'interno della città. Gli edifici vanno quindi importati singolarmente.

### 4.1 Struttura dei modelli

I modelli sono presenti all'interno di un archivio con organizzazione in cartelle X/Y/MODEL\_ID, dove X e Y sono le coordinate di una cella di firenze espresse secondo un sistema di riferimento standard, mentre MODEL\_ID è l'identificativo dei modelli presenti all'interno della cella. All'interno di ciascuna cartella sono presenti i modelli sia ad alta che a bassa risoluzione. Dato lo scaling automatico effettuato da UE5, solo i modelli ad alta risoluzione vengono utilizzati.

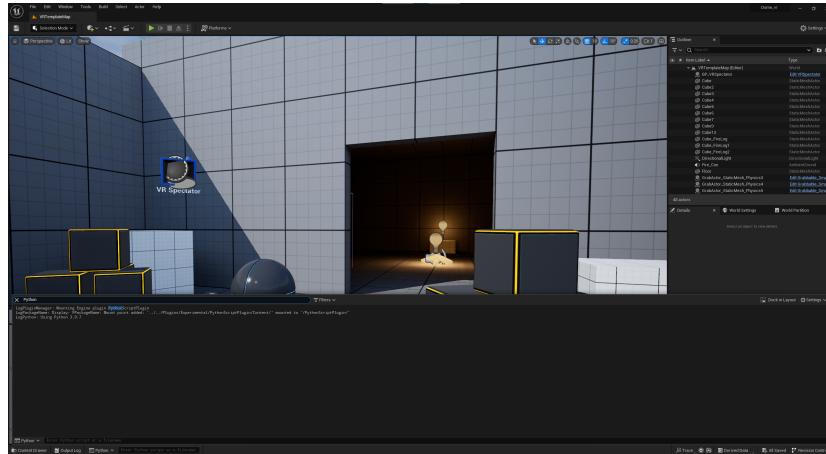
Dato che all'interno di ogni cella i modelli condividono la stessa texture (adeguatamente posizionata) ciò crea problemi al modo in cui UE effettua l'import dei modelli, che se importanti in una singola destinazione importano solamente la texture del primo elemento di ogni cella, lasciando gli altri senza texturizzazione. Per risolvere questo problema è possibile forzare l'import di ripetizioni della texture mantendendo la struttura dell'archivio anche all'interno di UE. Per fare ciò è stato prodotto lo script `import.py`, che impiega le API di UE per importare in modo automatico i file. La sintassi del comando è

```
import.py --start_x SX --start_y SY --end_x EX --end_y EY
```

dove gli argomenti richiesti vanno a definire le celle da importare, che corrispondono alle celle della che hanno coordinata  $X \in [SX, EX]$  e coordinata  $Y \in [SY, EY]$ . A causa del comportamento interno di UE, non è possibile usare negli script elementi caricati nell'editor ma non salvati, perciò è OBBLIGATORIO salvare tutti gli elementi importati al termine del processo. Tale processo richiede la compilazione degli shader associati ai modelli, ed essendo un task estremamente oneroso è da eseguire in batch contenuti, pena l'arresto imprevisto dell'editor, con perdita di tutti i progressi. Il procedimento è mostrato in figura 4 Una volta importati e salvati i modelli è possibile posizionarli all'interno del livello, procedura eseguibile tramite uno script addizionale, che evita di dover allocare i modelli singolarmente:

```
spawn.py --start_x SX --start_y SY --end_x EX --end_y EY --scale S
```

La sintassi è la stessa del comando `import.py`, ma con un parametro aggiuntivo che indica il fattore di scala da applicare ai modelli durante la creazione: può essere omesso, ed in tal caso la scala sarà 1 (grandezza naturale).



(a) Aprire la finestra OUTPUT LOG, e optionalmente digitare Python nella barra di ricerca in alto, in modo da filtrare gli output non pertinenti

```
Python
LogPluginManager: Mounting Engine plugin PythonScriptPlugin
LogPackageName: Display: FPackageName: Mount point added: '../../../../../Plugins/Experimental/PythonScriptPlugin/Content/' mounted to '/PythonScriptPlugin/'
LogPython: Using Python 3.9.7
```

(b) Nella barra dei comandi in basso passare da CMD a Python, ed eseguire lo script di PATH\_TO\_SCRIPT/import.py, specificandone la posizione del disco e i parametri richiesti

```
Python
LogPluginManager: Mounting Engine plugin PythonScriptPlugin
LogPackageName: Display: FPackageName: Mount point added: '../../../../../Plugins/Experimental/PythonScriptPlugin/Content/' mounted to '/PythonScriptPlugin/'
LogPython: Using Python 3.9.7
```

(c) Dopo aver salvato i modelli importati, eseguire lo script di PATH\_TO\_SCRIPT/spawn.py, specificandone la posizione del disco e i parametri richiesti

Figure 4: Procedura di import dei modelli all'interno di UE.

## 5 Terreno

Dopo aver caricato i modelli occorre fornire al mondo generato un terreno sul quale muoversi.

Il caricamento è derogabile a due processi differenti: caricamento della texture e caricamento della mappa altimetrica. Usare l'immagine altimetrica per creare un terreno somigliante a quello fiorentino non è stato possibile, in quanto allineare terreno e edifici richiede il caricamento di una porzione maggiore della città, cosa non possibile a causa dell'hardware usato, di limitata potenza.

### 5.1 Texture

La texture del terreno è ottenuta con la stessa logica della suddivisione della mappa in una griglia. Si effettua una chiamata al server arcgis per ottenere le patch di una zona in particolare per poi unirle successivamente. La chiamata è la seguente

```
https://server.arcgisonline.com/ArcGIS/rest/services/World\_Imagery/MapServer/tile/Z/Y/X
```

dove X, Y sono le coordinate nella griglia, e Z è il fattore di zoom, che per coerenza con gli altri componenti deve avere valore 18. L'esecuzione delle varie chiamate al server è gestita dallo script

```
get_patches.py --start_x SX --start_y SY --end_x EX --end_y EY
```

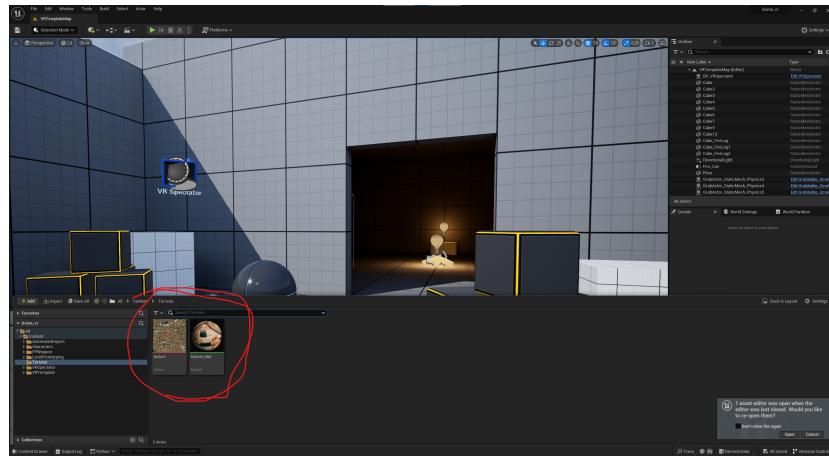
che salva le patch nella cartella "patches" contenuta nel percorso dello script, mentre con la chiamata a

```
create_texture.py
```

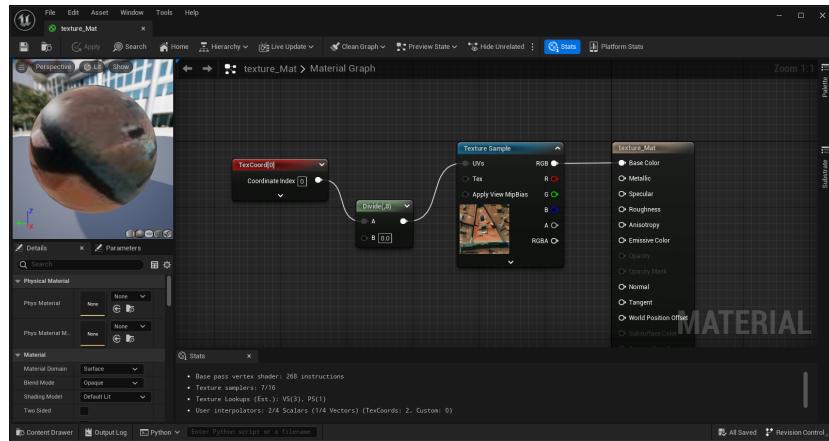
le varie patch verranno unite e salvate nel file `texture.png`. Per usare la texture va prima importata nell'editor, e per fare ciò basta trascinarla dal file explorer all'interno dell'editor: questo creerà un asset di tipo texture nel posto scelto. A questo punto è possibile trascinare la texture sull'oggetto da texturizzare, e facendo così verrà creato un `materiale` associato, che presenterà il pattern della texture ripetuto. Per evitare tale ripetizione bisogna effettuare un upsampling della texture, azione eseguibile all'interno del blueprint del materiale, consultabile cliccando due volte sul materiale. Nel blueprint bisogna aggiungere i nodi presenti in figura 5, creando quelli mancanti dal menu che appare cliccando col tasto destro del mouse.

### 5.2 Modello del terreno da scansione altimetrica

Nella modalità `Landscape` dell'editor è possibile generare un terreno a partire da un'immagine PNG in scala di grigi. Tale immagine è ottenibile tramite una chiamata al server di `snap4city` al seguente indirizzo



(a) Texture e materiale associato



(b) Il fattore di upsampling è determinato dal valore costante del blocco Divisione, aggiustare quello affinche non siano presenti ripetizioni.

Figure 5: Procedura di upsampling della texture del terreno.

```

https://wmsserver.snap4city.org/geoserver/wms?
service=WMS&request=GetMap&layers=TuscanyDTM1&styles=
&format=image%2Fpng&transparent=true&version=
1.1.1&width=WIDTH&height=HEIGHT&srs=EPSG%3A4326&bbox
=LONG1,LAT1,LONG2,LAT2

```

dove LONG1, LAT1, LONG2 e LAT2 rappresentano la bounding box dell'area geografica di interesse (è consigliato farla coincidere con un sottinsieme di celle della griglia usata fino a questo momento), metri HEIGHT e WIDTH rappresentano la risoluzione dell'immagine prodotta. Per essere compatibile con UE l'immagine deve essere quadrata, quindi HEIGHT = WIDTH, e non tutte le risoluzioni sono compatibili. La risoluzione consigliata è la più grande che è possibile inserire senza causare problemi all'editor, e nel nostro caso è di  $6033 \times 6033$ , però per seguire dei test può essere comodo creare di più piccole: una documentazione più esaustiva è presente nella guida tecnica ai paesaggi su UE 5.2.

## 6 Oculus

### 6.1 Set-up

#### Setup di Oculus Rift

##### **Passo 1: Verifica dei requisiti di sistema**

Prima di iniziare, assicurati che il tuo computer soddisfi i requisiti minimi di sistema per Oculus Rift. Puoi trovare questi requisiti sul sito web ufficiale di Oculus.

##### **Passo 2: Scarica l'app Oculus**

1. Vai sul sito web ufficiale di Oculus e scarica l'app Oculus per il tuo sistema operativo (Windows o macOS).
2. Esegui il file di installazione scaricato e seguì le istruzioni per completare l'installazione.

##### **Passo 3: Collegamento del visore Oculus Rift**

1. Collega il visore Oculus Rift al tuo computer utilizzando i cavi appropriati.

##### **Passo 4: Configurazione dei controller Oculus Touch**

1. Se hai i controller Oculus Touch, dovrà configurerli. Segui le istruzioni fornite nell'app Oculus per accoppiare e configurare i controller.

##### **Passo 5: Avvio dell'app Oculus e configurazione**

1. Avvia l'app Oculus sul tuo computer.

2. Segui le istruzioni visualizzate sullo schermo per configurare il tuo visore Oculus Rift. Questo potrebbe includere la creazione di un account Oculus, l'aggiornamento del firmware del visore e l'importazione delle tue impostazioni.

**Passo 6: Accedi al menu principale di Oculus**

1. Premi il pulsante Oculus sul tuo controller per accedere al menu principale di Oculus.

**Passo 7: Naviga nel menu principale di Oculus**

1. Utilizza il joystick o il touchpad sul controller Oculus per navigare nel menu principale.

**Passo 8: Abilita Oculus Link**

1. Per abilitare Oculus Link e collegare il visore Oculus Rift al tuo computer, vai su "Impostazioni" nel menu principale di Oculus.
2. Seleziona "Dispositivi" e quindi "Oculus Link" o "Cavo USB".
3. Segui le istruzioni visualizzate per completare la configurazione di Oculus Link.

**Passo 9: Accedi al desktop virtuale**

1. Una volta collegato al computer tramite Oculus Link, potrai accedere al desktop virtuale come se stessi usando il tuo computer monitor attraverso il visore Oculus Rift.
2. Utilizza il controller Oculus per navigare tra le applicazioni e il desktop virtuale.
3. Segui le istruzioni per creare una "Play Area" virtuale. Questa è l'area in cui puoi muoverti fisicamente mentre usi il visore. Assicurati che l'area sia priva di ostacoli.

## 6.2 Controls

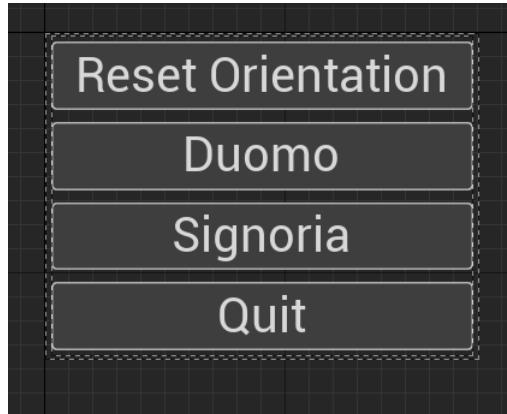
### 6.2.1 Game Menu

Il menu di gioco è stato creato seguendo questi passaggi:

1. Modifica del WidgetMenu pre-impostato
2. Creazione di più livelli in base ai punti di interesse
3. Locazione degli starting point in base ai punti di interesse

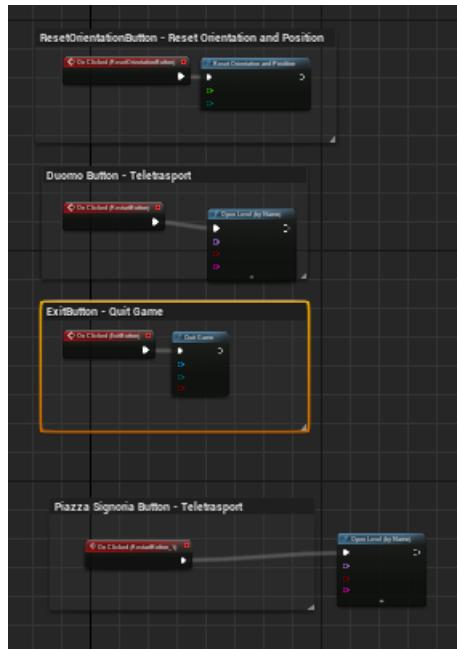
### Modifica del widget Menu:

Il widget menu che è pre-impostato, è stato modificato aggiungendo due tasti per il teletrasporto nei punti di interesse della mappa. Il menu si presenta così



- Reset orientation: resetta l'orientazione
- Duomo: ci teletrasporta di fronte al duomo
- Signoria: ci teletrasporta in piazza signoria di fronte a palazzo vecchio
- Quit: spegne il gioco

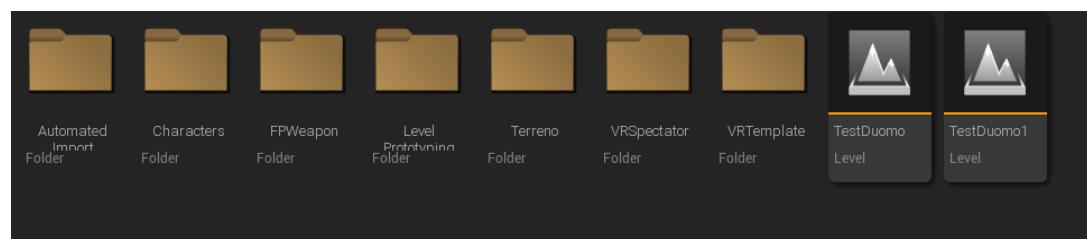
Questa è la logica dietro i bottoni creati:



**Creazione di più livelli in base ai punti di interesse:** Per permettere il teletrasporto tra punti diversi della città sono possibili due soluzioni:

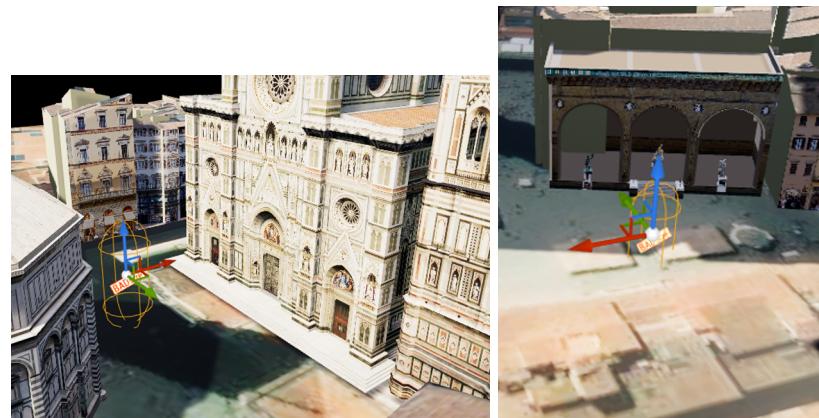
- aggiungere più punti di spawn all'interno del livello.
- creare più livelli con spawn differenti.

In prospettiva di un'estensione della mappa, la seconda alternativa è preferibile, in quanto consente di ridurre il carico di lavoro dell'oculus, quindi abbiamo perseguito tale approccio. Per permettere il teletrasporto quindi è stato generato un nuovo livello in cui si è settato un nuovo starting point, così quando si va a cliccare, per esempio, il bottone Duomo, ci viene caricato un nuovo livello con quel determinato starting point (i tempi di caricamento sono brevi se il livello è ragionevolmente contenuto). I due livelli TestDuomo e TestDuomo1 sono quelli definiti nell'immagine sottostante



#### **Locazione degli starting point:**

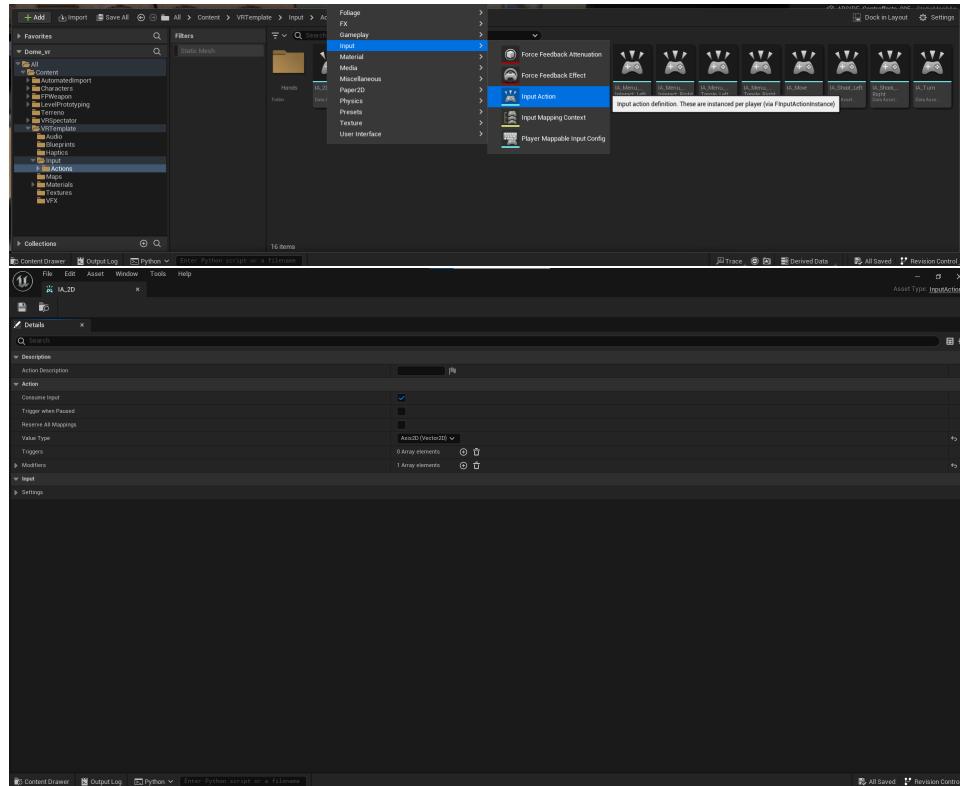
Come già detto abbiamo localizzato gli starting point in base ai punti di interesse in questo modo:



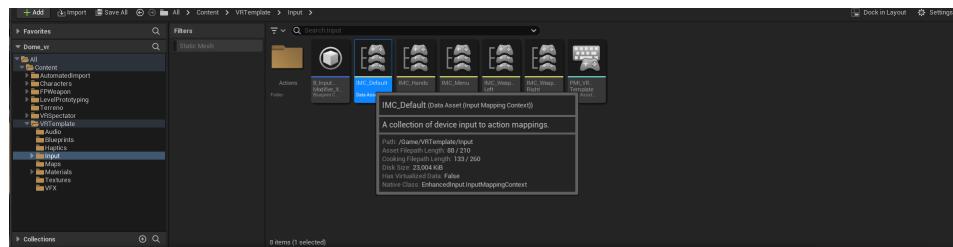
### 6.2.2 Character movement

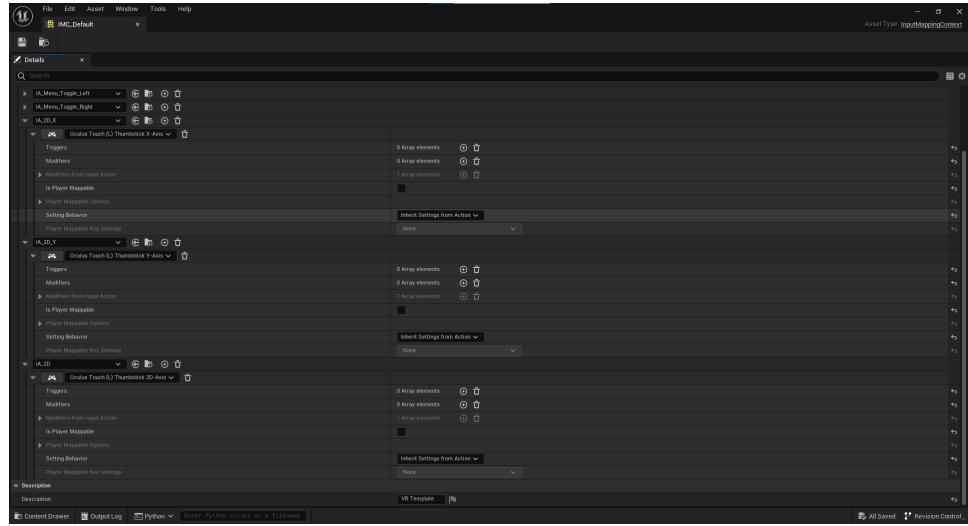
Il movimento del personaggio, implementata ovviamente anche questa su Unreal, segue 3 step:

1. Creazione dell'evento di movimento: dal content drawer aprire la cartella VRTemplate/Input/Actions e tramite click destro creare un oggetto di tipo "Input/Action"

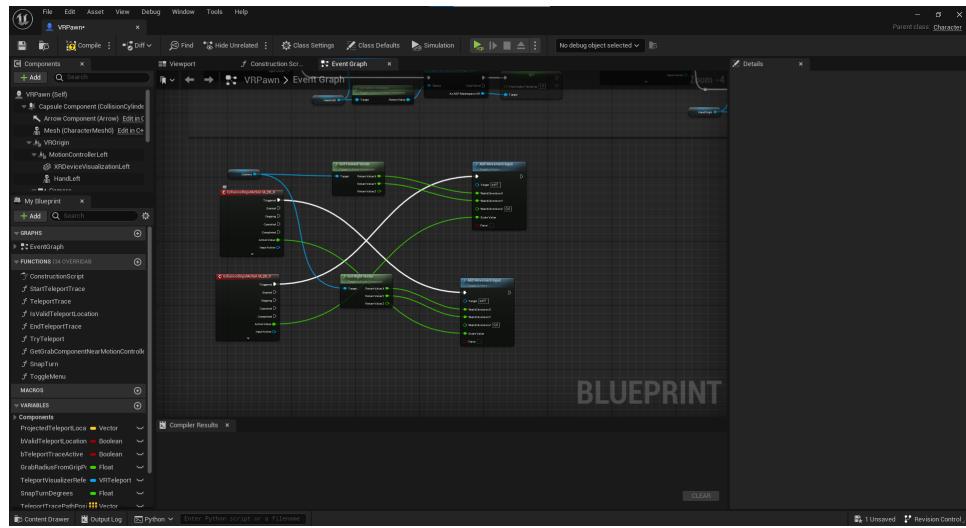


2. Associazione di tale evento al controller che consiste nell'associare ad ogni l'azione che si vuole eseguire a determinati tasti del controller. Il controller si trova nel content drawer





3. Implementazione della logica di base per il movimento che consiste nel modificare il blueprint del personaggio, ovvero il codice per far sì che esegua le azioni richieste nel modo giusto



## 7 Facade texturization

La facade texturization è stata eseguita su modelli 3D del lungarno di Firenze e consiste, in breve, nell'applicare facciate prese da foto in alta risoluzione correttamente rettificate e ritagliate. Il processo si può dividere in tre step principali:

1. Riconoscimento degli edifici
2. Modellazione delle foto
3. Applicazione delle facciate

Gli edifici sono stati divisi in cartelle numerate in base ad un id che li identifica. La struttura delle cartelle è di questo tipo:

Coordinate orizzontali

139190	09/03/2023 18:59	Cartella di file
139191	09/03/2023 18:59	Cartella di file
139192	09/03/2023 18:59	Cartella di file
139193	09/03/2023 18:59	Cartella di file
139194	09/03/2023 18:59	Cartella di file
139195	09/03/2023 18:59	Cartella di file
139196	09/03/2023 18:59	Cartella di file
139197	09/03/2023 18:59	Cartella di file
139198	09/03/2023 18:59	Cartella di file
139199	09/03/2023 18:59	Cartella di file
139200	09/03/2023 18:59	Cartella di file
139201	09/03/2023 18:59	Cartella di file

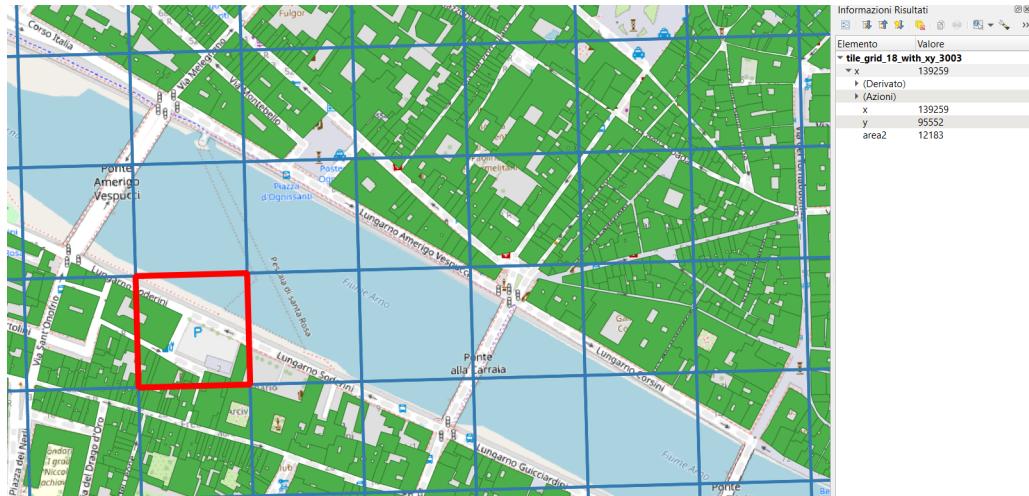
Coordinate verticali

95523	09/03/2023 18:59	Cartella di file
95524	09/03/2023 18:59	Cartella di file
95525	09/03/2023 18:59	Cartella di file
95526	09/03/2023 18:59	Cartella di file
95527	09/03/2023 18:59	Cartella di file
95528	09/03/2023 18:59	Cartella di file
95529	09/03/2023 18:59	Cartella di file
95530	09/03/2023 18:59	Cartella di file
95531	09/03/2023 18:59	Cartella di file
95532	09/03/2023 18:59	Cartella di file
95533	09/03/2023 18:59	Cartella di file
95534	09/03/2023 18:59	Cartella di file
95539	09/03/2023 18:59	Cartella di file
95540	09/03/2023 18:59	Cartella di file
95541	09/03/2023 18:59	Cartella di file
95542	09/03/2023 18:59	Cartella di file
95543	09/03/2023 18:59	Cartella di file
95544	09/03/2023 18:59	Cartella di file
95545	09/03/2023 18:59	Cartella di file

## Id modelli

I vari step sono stati eseguiti in questo modo:

1. Utilizzo di QGIS per trovare corrispondenza tra edifici e cartelle



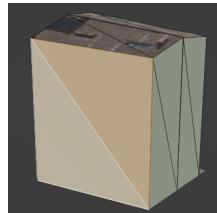
Con la griglia si possono selezionare i singoli edifici della mappa e ricavare le coordinate. Una volta fatto questo bisogna capire quali foto corrispondono all'edificio utilizzando google street view per esempio

2. Una volta capito quali foto corrispondono ai vari edifici si importano in un programma di editing per eseguire la rettificazione, il ritaglio e il resize dell'immagine



3. Per l'applicazione delle foto sulle facciate si è utilizzato Blender insieme ad un plugin per velocizzare il tutto. Il processo in breve si divide nei seguenti passaggi:

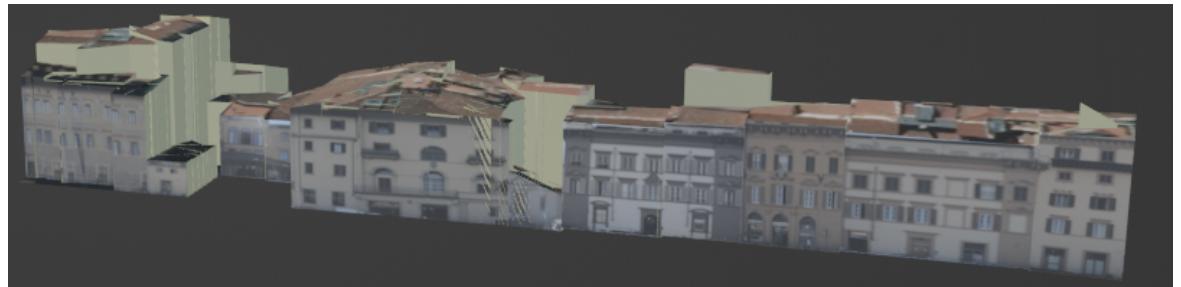
- Importo su blender il modello scelto durante lo step 1
- Seleziono la facciata su cui applicare l'immagine



- Eseguo il plugin per la facade texturization



Esempio di pezzo di lungarno texturizzato



### **Tips per evitare errori e velocizzare il processo:**

- Una volta che si importa un modello su blender non spostarlo ma selezionare la vista tramite view → frame selected altrimenti il plugin non funziona correttamente
- Per velocizzare l'applicazione di textures importare diversi modelli adiacenti in modo da capire più velocemente quale immagine (o immagini) corrisponde ad un determinato edificio
- Utilizzare google street view se non è chiara la posizione degli edifici

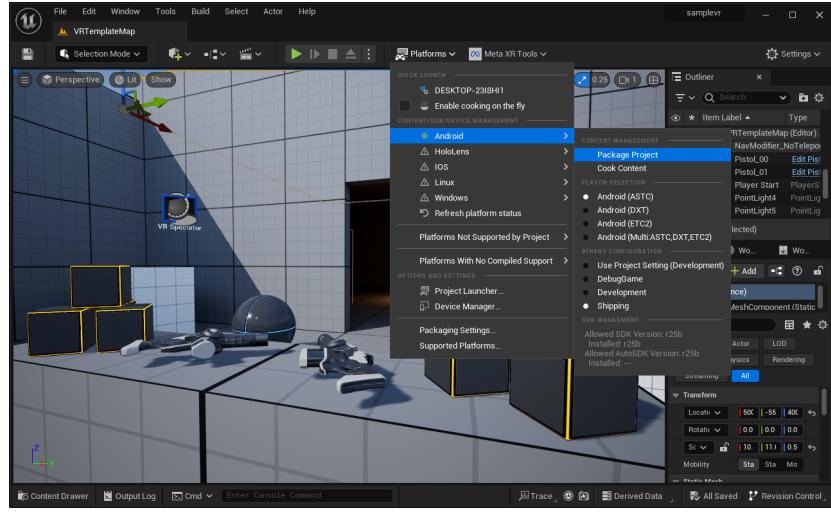
## **8 Compilazione e Installazione su Oculus**

### **8.1 Abilitare il supporto alla compilazione**

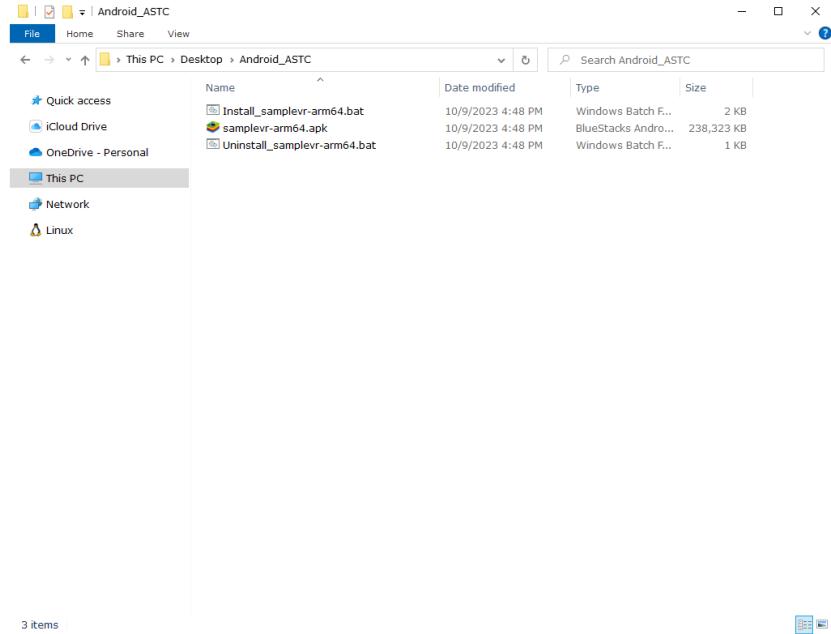
Gli eseguibili per oculus quest hanno lo stesso formato delle applicazioni Android (APK), e richiedono dei componenti aggiuntivi per poter essere creati:

- Android studio 4.0: reperibile nel sito ufficiale e nel materiale allegato ([installer/android-studio-ide-193.6514223-windows.exe](#)), permette di installare il Software Developer Kit (SDK) 29 richiesto per la compilazione, insieme a il kit di Command Line Tools 8.0, entrambi installabili dall'SDK manager di Android studio. Insieme a questi componenti viene installata anche la Java Runtime Environment 8, necessaria per creare il file APK.
- Meta XR plugin per UE 5.2, reperibile tra i file allegati, nella posizione [plugins/UnrealMetaXRPlugin.56.0.zip](#). L'installazione deve seguire quanto indicato in <https://developer.oculus.com/downloads/package/unreal-engine-5-integration/>. Il plugin deve essere abilitato dal menu Edit → Plugins, seguendo la medesima procedura riportata in figura 3.

Per preparare il progetto alla compilazione seguire i passaggi riportati al link <https://developer.oculus.com/documentation/unreal/unreal-quick-start-guide-quest/>. Dopo aver eseguito questi passaggi è possibile compilare il progetto attraverso il menu Platforms →Android →ASTC →Package Project, selezionando la modalità SHIPPING per permettere al compilatore di ottimizzare il codice. Dopo aver selezionato la cartella di destinazione il processo di compilazione inizierà, creando i seguenti file: almeno un file APK (in base al progetto e al tipo di compilazione potrebbero essere di più), un eseguibile per installare il programma su oculus, e un eseguibile per rimuoverlo. Se il processo di compilazione termina con un errore, installare separatamente JRE 8 (installer fornito tra gli allegati) e specificare il path della runtime nel menu Edit →Project Settings. Un esempio di compilazione è riportato in figura 6



(a) Selezionando la modalità SHIPPING il progetto viene ottimizzato, rimuovendo i simboli di debug, operazione che migliora le prestazioni.



(b) Il file APK non ha un utilizzo manuale: per installarlo (o rimuoverlo) sull'oculus occorre utilizzare lo script generato.

Figure 6: Procedura di compilazione dell'applicazione.

## 8.2 Abilitare il debug usb nell'oculus

Per permettere l'installazione dell'applicazione su oculus occorre attivare la modalità sviluppatore. Assumendo che l'oculus abbia come account amministratore un account abilitato allo sviluppo (l'account è già presente, ma in caso servisse crearne uno nuovo la registrazione come developer si può fare da questo sito <https://developer.oculus.com/resources/publish-account-management-intro/>), accedere a tale account sull'applicazione per telefono Meta Quest (Android) o Oculus (iPhone), e connettere entrambi i dispositivi alla stessa rete WiFi. Dall'applicazione attivare la modalità debu attraverso  
Menu → Dispositivi → {OCULUS INTERESSATO} → Impostazioni Dispositivo → Modalità Sviluppatore → Debug USB.

## 8.3 Installazione dell'APK

Una volta eseguiti i passaggi precedenti, l'installazione dell'applicazione può avvenire da un qualsiasi computer, anche se non corrisponde a quello usato per lo sviluppo. Occorre installare sul computer usato per l'installazione dell'applicazione i driver di debug oculus (fortiniti tra gli allegati), e verificare che tutto funzioni tramite il comando per terminale `adb devices`, che in caso di esito positivo dovrebbe mostrare il codice di debug dell'oculus affiancato dalla scritta `authorized` (se l'accesso è stato garantito dal pop-up che appare alla connessione dell'oculus al computer). Per installare i driver è sufficiente aprire il context menu di windows (click destro) sul file `android_winusb.inf`, locato nella cartella `installers/oculus-adb-drivers-2.0/oculus-go-adb-drivers-2.0/usb_driver`, e selezionare `installa`.

A questo punto per installare l'applicativo occorre eseguire il file eseguibile prodotto dal processo di compilazione descritto in precedenza, che eseguirà il processo in modo automatico.