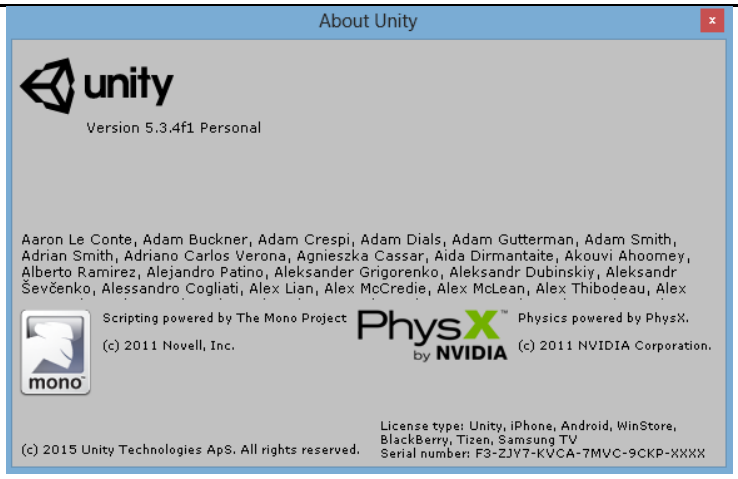
Cardboard VR Walking and Running

1. GENERAL INFORMATION

DATE OF DOCUMENT	30/04/2016
NAME OF THE PROJECT	Cardboard VR Walking and Running
AUTHOR	Michael Soler
UNITY VERSION	5.3.4.F1 PERSONAL



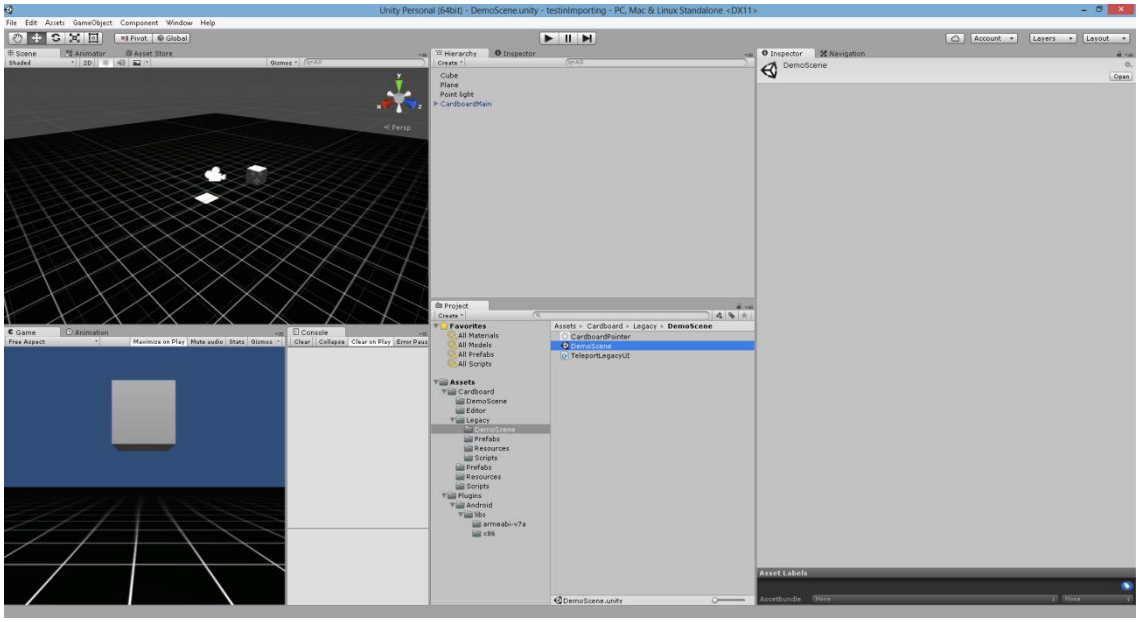
Index

2. IMPORTING INFORMATION

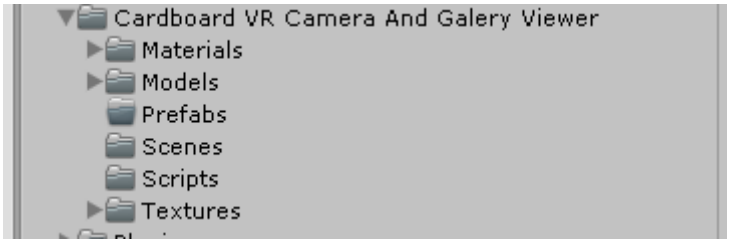
This package works with the "google cardboard" for UNITY that must be downloaded first using the following link:

https://developers.google.com/cardboard/unity/?hl=en

Once downloaded and imported to unity, your project should look like this:



Then, import our package to the project, which will leave you the following folder configuration:



The gaze input collider must be disabled. There is no need to change collider or other game objects.

3. PROJECT DESCRIPTION

This package allows the user to move in a virtual space without the need of gamepads or similar. The movement is obtained analyzing the input of the accelerometer in the Y (vertical) direction. The movement of the hips while simulating a walk generates a similar response to a sinusoid response. This function is used to determine the movement and its speed. To sum up, with this package you will be able to walk, run and jump with any king of hardware but your phone.

This package includes:

- Sample scene that works on Android.
- Movement Script that can be totally configurable (walking time, speed, etc).
- Sample textures and environnement.
- Complete documentation.

This package works with the "google cardboard" SDK=0.6 for UNITY that must be downloaded first using the following link:

https://developers.google.com/cardboard/unity/?hl=en

4. LAYERS, TAGS AND COLLIDERS

LAYERS

All objects are placed in the default layer.

TAGS:

Gameobjects are untagged.

COLLIDERS

- Box colliders are used for the environment.
- Capsule collider is used for the player (cardboard object).

Check if the "gaze pointer cursor" collider is disabled:

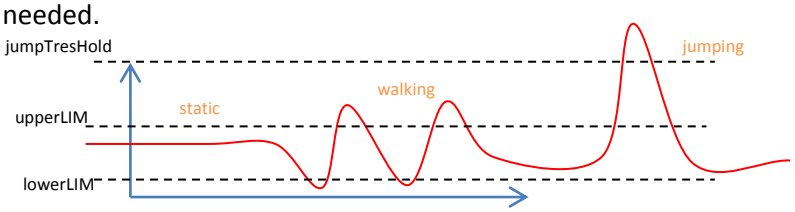
It is important to import the cardboard package correctly, and check if these colliders are working properly.

5. SCRIPTING INFORMATION

We explain each script with some detail in the following table:

- **WalkFilter.cs:**

  This script controls the movement on the Virtual world of the player. Most of the parameters are given as standard values, but user can change them if needed.



  The previous chart shows the evolution of the measurement of the accelerometer in the Y direction. This script uses some thresholds and limits to determine if the player is moving or he is jumping.

IMPORTANT VARIABLES
public Text x,y,z,vT; → They are used to display on a canvas the values of acceleration (testing). **public float lowerLiM=-1.3f;** → It is used to determine the lower acceleration to consider for the walking/running event. **public float upperLiM=-0.7f;** → It is used to determine the higher acceleration to consider for the walking/running event. **public float walkMinTime=1.5f;** → It is used to reset the timer values. **public float speedFactor=0.25f;** → It is used to adapt the real movement to the virtual one. **public float jumpTresHold=-0.5f;** → It is used to determine it the player has performed a jump. **public Transform head;** → It is the reference to the head transform. **public float JumpSpeed=250f;** → It is the coefficient used to introduce the necessary force for the jump. **public float timeJump=0.7f;** → It is amount of time between jumps (needed to prevent duplicates). **public float deltaDerivative=0.05f;** → It is used to obtain the derivative of the acceleration. **public float derivativeThreshold=20;** → It is to trigger a jump. **public float Dacc;** → It is the value of the acceleration derivative.
IMPORTANT FUNCTIONS
public void move(float v) → It is used to move the player a "v" amount of distance. **public void jump()**→ It is used to simulate the jump. **public void stopJump()**→ It is used to set speed to zero (in case of existing residual speed / other model problems).

Other scripts are commented on the “*.cs” files.