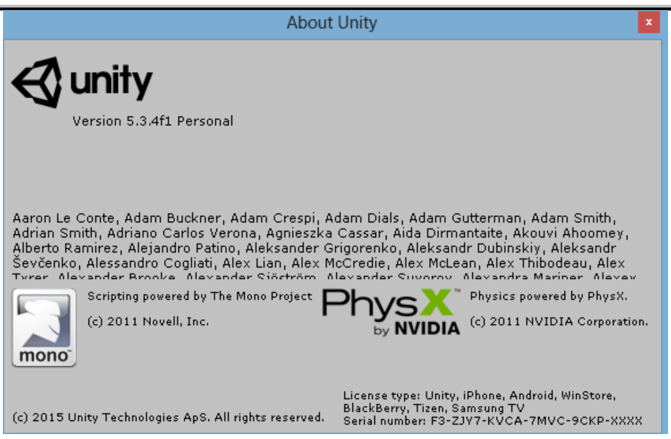
# Cardboard VR FPS basic kit

## 1. GENERAL INFORMATION

| | |
|---|---|
| **DATE OF DOCUMENT** | 4/4/2016 |
| **NAME OF THE PROJECT** | Cardboard VR FPS basic kit |
| **AUTHOR** | Michael Soler |



Version 5.3.4f1 Personal

Aaron Le Conte, Adam Buckner, Adam Crespi, Adam Dials, Adam Gutterman, Adam Smith, Adrian Smith, Adriano Carlos Verona, Agnieszka Cassar, Aida Dirmantaite, Akouvi Ahoomey, Alberto Ramirez, Alejandro Patino, Aleksander Grigorenko, Aleksandr Dubinskiy, Aleksandr Ševčenko, Alessandro Cogliati, Alex Lian, Alex McCredie, Alex McLean, Alex Thibodeau, Alex Tyrer, Alexander Brooke, Alexander Sjöström, Alexander Suvorov, Alexandra Mariner, Alexey

Scripting powered by The Mono Project

(c) 2011 Novell, Inc.

Physics powered by PhysX.

(c) 2011 NVIDIA Corporation.

(c) 2015 Unity Technologies ApS. All rights reserved.

License type: Unity, iPhone, Android, WinStore, BlackBerry, Tizen, Samsung TV
Serial number: F3-ZJY7-KVCA-7MVC-9CKP-XXXX

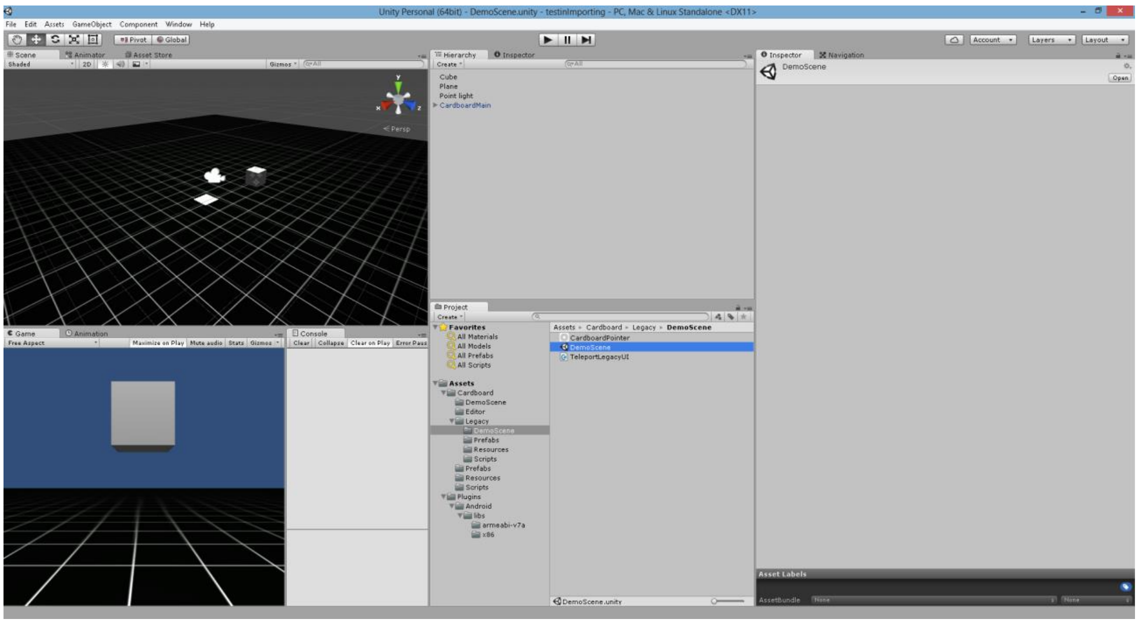
## Index

## 2. IMPORTING INFORMATION

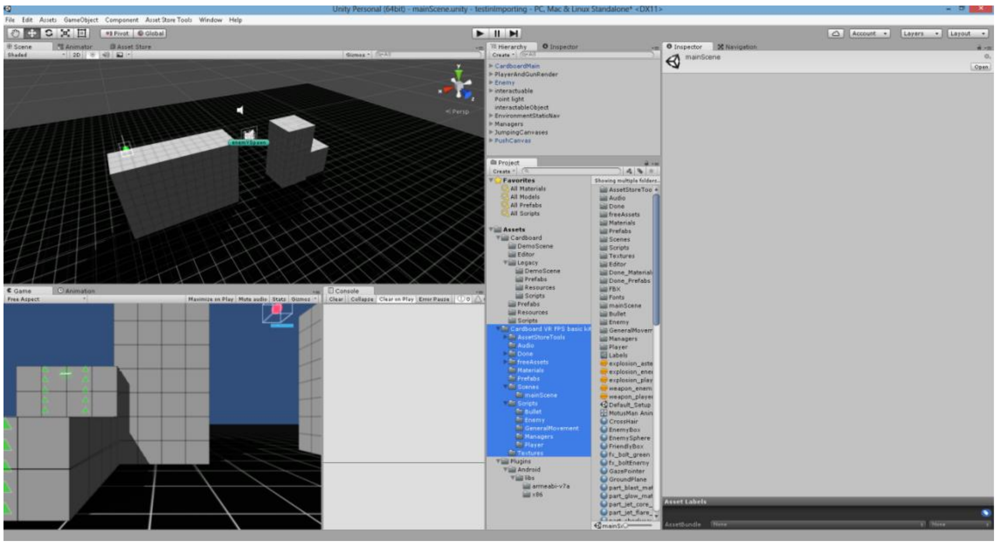
**This package works with the "google cardboard" for UNITY that must be downloaded first using the following link:**

**https://developers.google.com/cardboard/unity/**

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



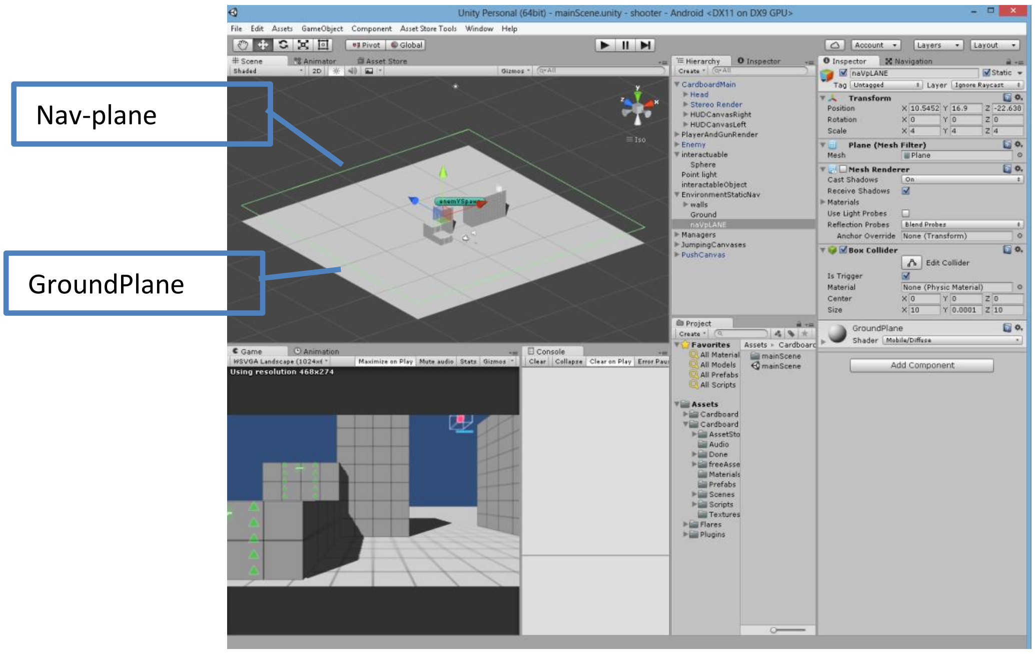
**Then, import our package to the project, which will leave you the following configuration:**



There is no need to change collider or other game objects.

## 3. PROJECT DESCRIPTION

The current application uses a navmesh that is positioned on top of the player and it is transparent. The enemies move on this plane.  When enemy health is equal to zero, the gameobject falls to the player surface.


Nav-plane
GroundPlane
Unity Personal (64bit) - mainScene.unity - shooter - Android <DX11 on DX9 GPU>
File Edit Assets GameObject Component Asset Store Tools Window Help
Pivot Global
Account Layers Layout
Scene Animator Asset Store
Shaded 2D Gizmos
Game Animation
Console
Clear Collapse Clear on Play Error Pause
Maximize on Play Mute audio Stats Gizmos
Using resolution 468x274
Add Component

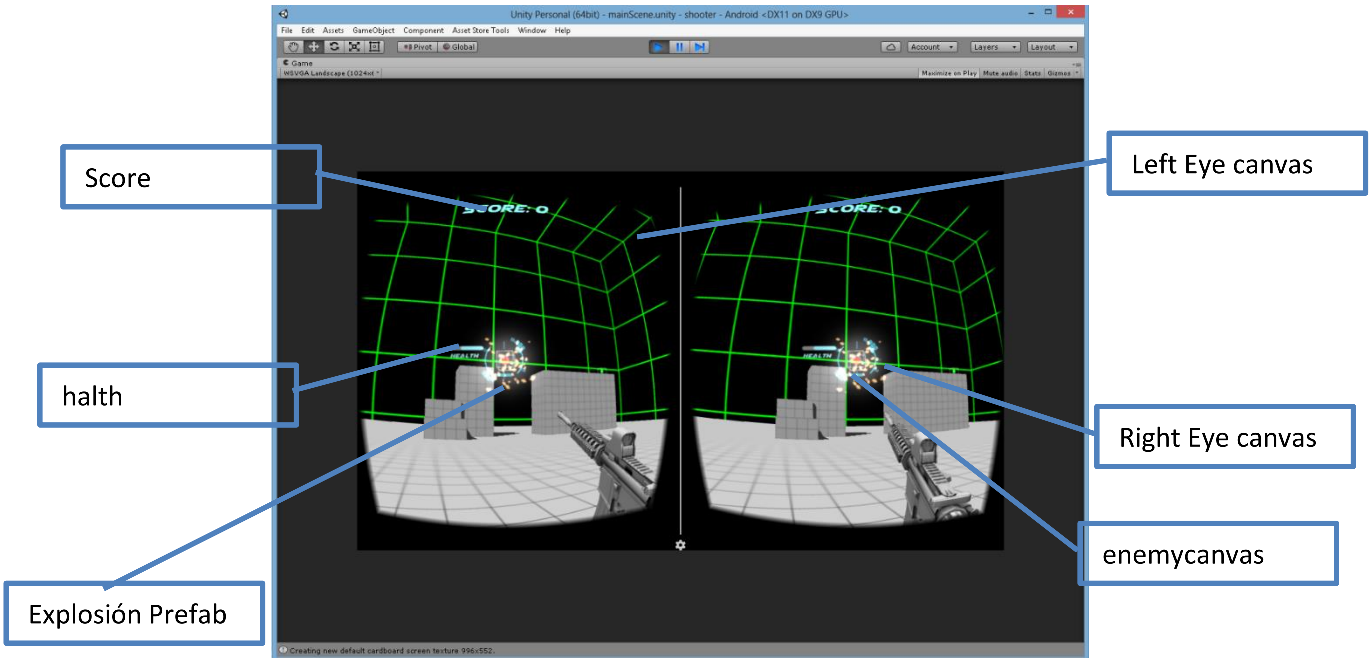
```
Hierarchy
CardboardMain
  Head
  Stereo Render
  HUDCanvasRight
  HUDCanvasLeft
PlayerAndGunRender
Enemy
interactuable
  Sphere
Point light
interactableObject
EnvironmentStaticNav
  walls
  Ground
  naVpLANE
Managers
JumpingCanvases
PushCanvas

Inspector
naVpLANE                Static
Tag Untagged            Layer Ignore Raycast
Transform
  Position   X 10.5452  Y 16.9    Z -22.638
  Rotation   X 0        Y 0       Z 0
  Scale      X 4        Y 4       Z 4
Plane (Mesh Filter)
  Mesh       Plane
Mesh Renderer
  Cast Shadows        On
  Receive Shadows     ✓
  Materials
  Use Light Probes    ☐
  Reflection Probes   Blend Probes
  Anchor Override     None (Transform)
Box Collider
  Edit Collider
  Is Trigger  ✓
  Material    None (Physic Material)
  Center     X 0        Y 0       Z 0
  Size       X 10       Y 0.0001  Z 10
GroundPlane
  Shader  Mobile/Diffuse

Project
Favorites
  All Material
  All Models
  All Prefabs
  All Scripts
Assets
  Cardboard
  Cardboard
    AssetSto
    Audio
    Done
    freeAsse
    Materials
    Prefabs
    Scenes
    Scripts
    Textures
  Flares
  Plugins
Assets ▸ Cardboard
  mainScene
  mainScene
```

The player moves on the ground plane, which has colliders and rigidbody constrains to prevent unrealistic movements. When the player dies, constrains are changed to simulate the player's falling when dying. The game restarts automatically.

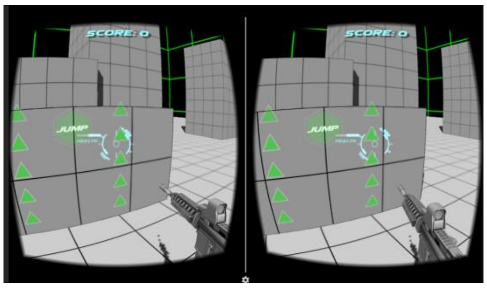
There is also an EnemyManager and ScoreManager which update the information on the HUDcanvas. This canvas is duplicated, as it is needed for VR applications. VR applications are characterized for using two cameras that are used for the left and right eyes. Those cameras are used as a reference for the HUD canvas that contains the score, crosshair and health. The rotation of the head inside the cardboard object is applied to the player object (that contains the gun).


Unity Personal (64bit) - mainScene.unity - shooter - Android <DX11 on DX9 GPU>
Score
Left Eye canvas
halth
Right Eye canvas
enemycanvas
Explosión Prefab
SCORE: 0
HEALTH
Creating new default cardboard screen texture 996x552.

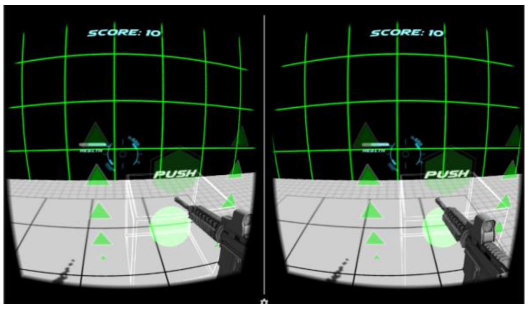
The user can shoot the enemies and a particle system plays a short explosion. The enemy health is reduced a specific amount in each hit. A canvas is also used to display enemy health.

Finally the player can interact with the environment thanks to several canvas on the objects, that execute some actions:

Jumping



Pushing



## 1. LAYERS, TAGS AND COLLIDERS

LAYERS

All objects are placed on the default layer.

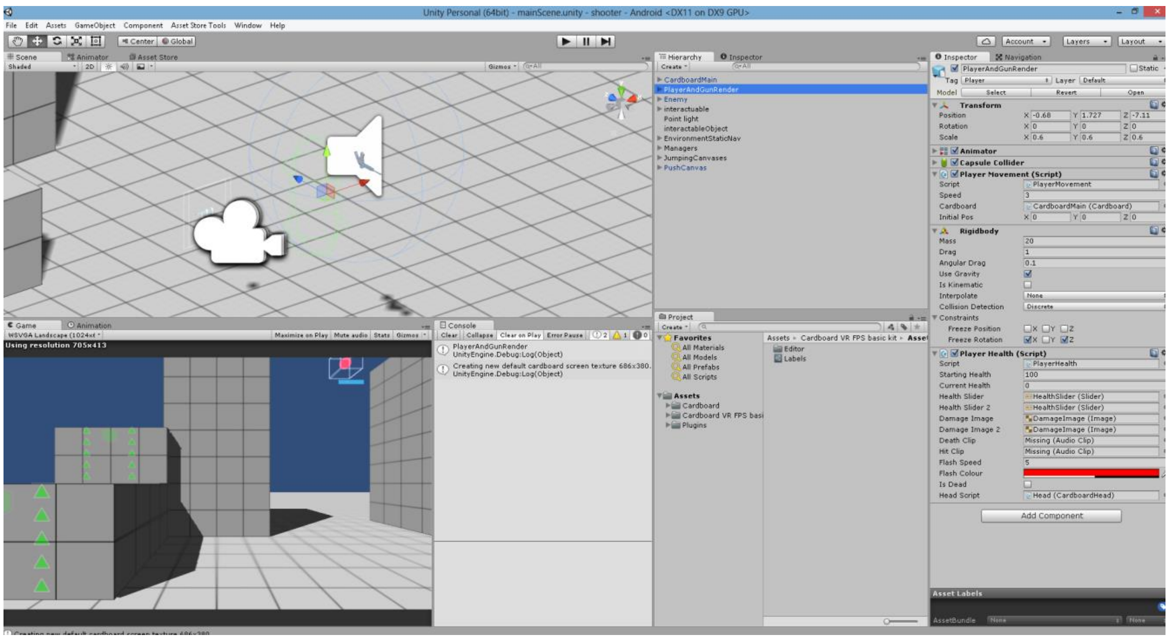
**TAGS:**

| TAG | Player | enemy |
|---|---|---|
| GAMEOBJECT | PlayerAndGunRender | Enemy |

*Other gameobjects are untagged.

COLLIDERS

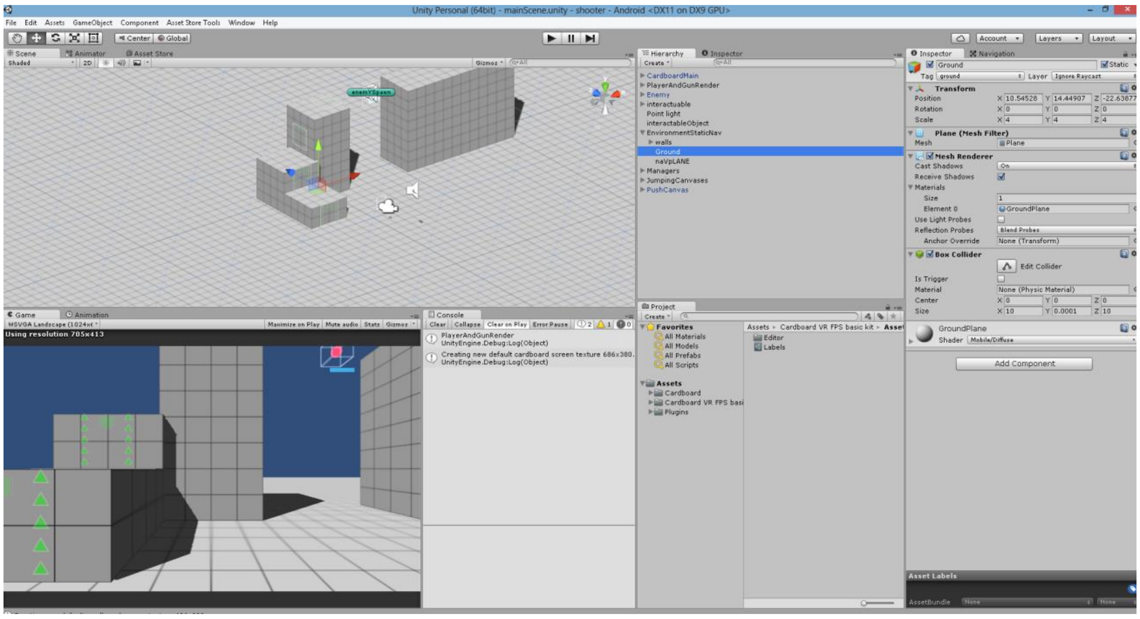
The “PlayerAndGunRender” has the main collider, which prevents the player from falling.



```
PlayerAndGunRender
UnityEngine.Debug:Log(Object)
Creating new default cardboard screen texture 686x380.
UnityEngine.Debug:Log(Object)
```

```
PlayerAndGunRender                 Static
Tag  Player        Layer  Default
Transform
Position          X -0.68   Y 1.727   Z -7.11
Rotation          X 0       Y 0       Z 0
Scale             X 0.6     Y 0.6     Z 0.6
Animator
Capsule Collider
Player Movement (Script)
Script            PlayerMovement
Speed             3
Cardboard         CardboardMain (Cardboard)
Initial Pos       X 0       Y 0       Z 0
Rigidbody
Mass              20
Drag              1
Angular Drag      0.1
Use Gravity       ✓
Is Kinematic      ☐
Interpolate       None
Collision Detection  Discrete
Constraints
  Freeze Position   ☐X ☐Y ☐Z
  Freeze Rotation   ✓X ☐Y ✓Z
Player Health (Script)
Script            PlayerHealth
Starting Health   100
Current Health    0
Health Slider     HealthSlider (Slider)
Health Slider 2   HealthSlider (Slider)
Damage Image      DamageImage (Image)
Damage Image 2    DamageImage (Image)
Death Clip        Missing (Audio Clip)
Hit Clip          Missing (Audio Clip)
Flash Speed       5
Flash Colour      🟥
Is Dead           ☐
Head Script       Head (CardboardHead)
```

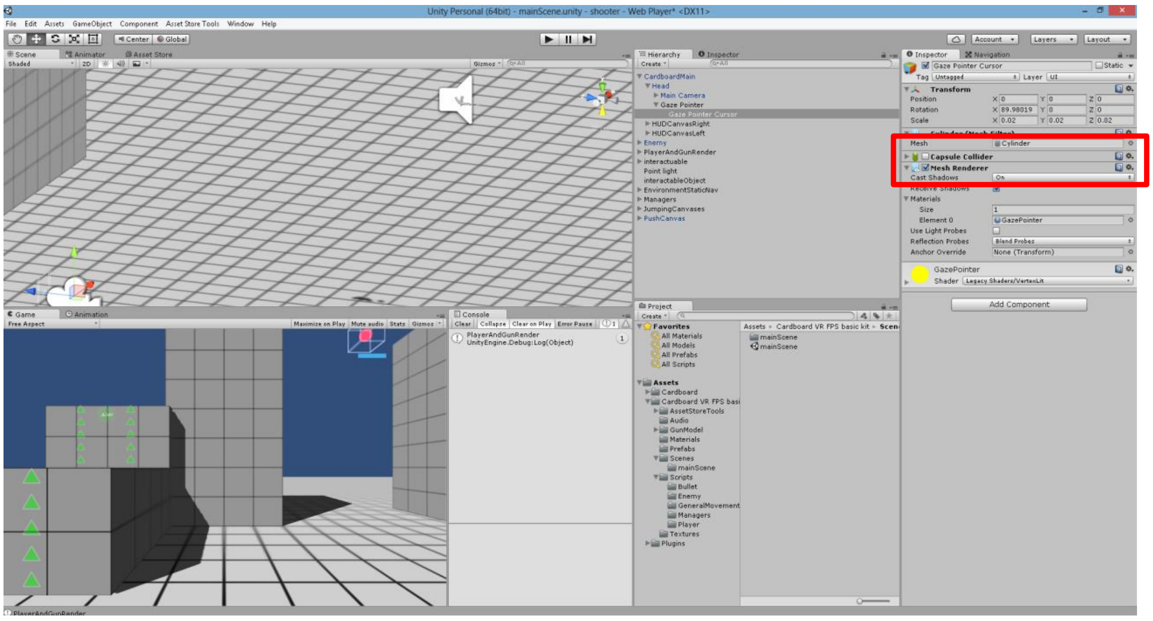
The ground has also a plane collider:



PlayerAndGunRender
UnityEngine.Debug:Log(Object)

Creating new default cardboard screen texture 686x380.
UnityEngine.Debug:Log(Object)

Check if the “gaze pointer cursor” collider is disabled:



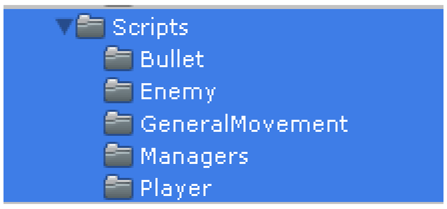
PlayerAndGunRender
UnityEngine.Debug:Log(Object)

NOTE

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

## 2. SCRIPTING INFORMATION

Scripting is sub-divided into the following folders:



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

- Bullet:
  - BulletMovement.cs

    This script controls the movement and color of the bullet:

| IMPORTANT VARIABLES |
|---|
| **bool isEnemy** → changes the material that is used. (Red=enemy, Green=player). |
| IMPORTANT FUNCTIONS |
| **OnCollisionEnter()** → according to the "tag" of the hit object, the code executes some specific commands, such as health reduction. |

- Enemy:
  - EnemyAttack.cs

    This script controls the attack capabilities of the enemies:

| IMPORTANT VARIABLES |
|---|
| **bool playerInRange**→ is used to know if the player is inside attack range. |
| IMPORTANT FUNCTIONS |
| **Attack ()** → reduces the player's health a given value |

  - EnemyHealth.cs

    This script manages the health of the enemies:

| IMPORTANT VARIABLES |
|---|
| **bool isDead**→ is used to know if the enemy is dead (health <=0) |
| IMPORTANT FUNCTIONS |
| **death ()** → changes the collision characteristics to leave the navMesh plane and triggers the autoDestruct() function.<br>autoDestruct() → is used to destroy the object after certain time. |

  - EnemyMovement.cs

    This script manages the movement of enemies.

| IMPORTANT VARIABLES |
|---|
| **float minDistanceToPlayer**→ distance that determines if the enemy is in too close to player.<br>**Float maxDistanceToPlayer** → distance that determines that the enemy is |

| |
|---|
| too far from the player. |
| IMPORTANT FUNCTIONS |
| FixedUpdate() → changes the position between both ranges, minDistanceToPlayer and maxDistanceToPlayer. |

- EnemyShooting.cs

  This script manages the creation of the bullets when shooting.

| IMPORTANT VARIABLES |
|---|
| **Float damagePerShot**→ the damage that a shot produces.<br>**timeBetweenBullets**→ the time needed between shots. |
| IMPORTANT FUNCTIONS |
| **Shoot()** → creates an instance of the bullet with the enemy proprieties. |

- GeneralMovement:
  - InteractionCanvas.cs

  This script makes the player jump and interact with objects when looking to a button/canvas. This is VR oriented.

| IMPORTANT VARIABLES |
|---|
| |
| IMPORTANT FUNCTIONS |
| **makePlayerJump()** → applies a velocity/force to make the player jump (up and forward).<br>**pushObject()** → applies a force/velocity to rigid body that contains the canvas. |

  - Look.cs

  This script makes the canvas face the player at all time.

| IMPORTANT VARIABLES |
|---|
| **Gameobject player** → the player that is on the scene. |
| IMPORTANT FUNCTIONS |
| |

- Managers:
  - EnemyManager.cs

  Creates enemies over the time.

| IMPORTANT VARIABLES |
|---|
| **Gameobject enemy** → the enemy prefab. |
| IMPORTANT FUNCTIONS |
| **deleteAllEnemies()** → removes enemies from game scene when called. |

- GameOverManager.cs

Restarts the application when game is ended (player.health<=0)

| IMPORTANT VARIABLES |
|---|
| **PlayerHealth playerhealth** → the player's health script. |
| IMPORTANT FUNCTIONS |
| |

- ScoreMangaer.cs

Updates the score while playing and destroying enemies.

| IMPORTANT VARIABLES |
|---|
| **Static int score** → the player's score. |
| IMPORTANT FUNCTIONS |
| **onGUI()** → is used to change the text of the canvas text objects. |

- Player:
  - PlayerHeath.cs

    This script manages the health of the player:

| IMPORTANT VARIABLES |
|---|
| **float currentHealth**→ is used to have the health evolution over time. |
| IMPORTANT FUNCTIONS |
| **death ()** → changes collision type and triggers the end of the game.<br>**restartLevel()** → starts the scene again once called. |

  - PlayerMovement.cs

    This script manages the movement of the player.

| IMPORTANT VARIABLES |
|---|
| **float speed**→ movement speed |
| IMPORTANT FUNCTIONS |
| **move()** → uses the keyboard input to move the player. It also calls to some cardboard functions in order to get the rotation angle. |

  - PlayerShooting.cs

    This is script is used to make the player shoot.

| IMPORTANT VARIABLES |
|---|
| **GameObject refNozzle**→ the point where the bullet will be crated.<br>**GameObject bullet** → the bullet prefab. |
| IMPORTANT FUNCTIONS |
| **Shoot ()** → It makes an Instance of the bullets. |