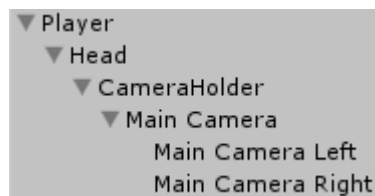




**Version 2.0**

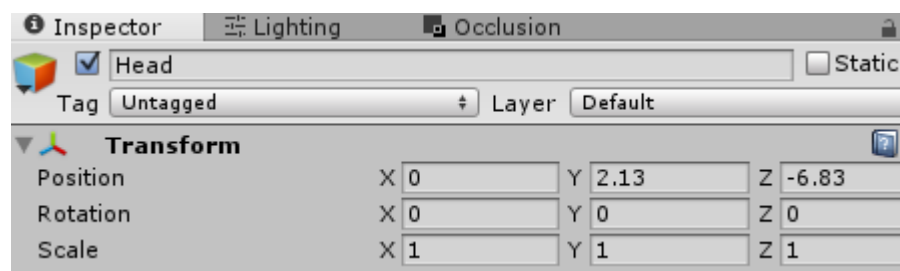
### **Installation for Oculus Gear VR / Google Cardboard (Default)**

1. Switch to Android platform in the build settings.
  2. For Google Cardboard you need to use Cardboard SDK and Google VR unity package in your project from Google developer
- 2.1. After integration of Google Cardboard package you need to comply with the following structure in the hierarchy window

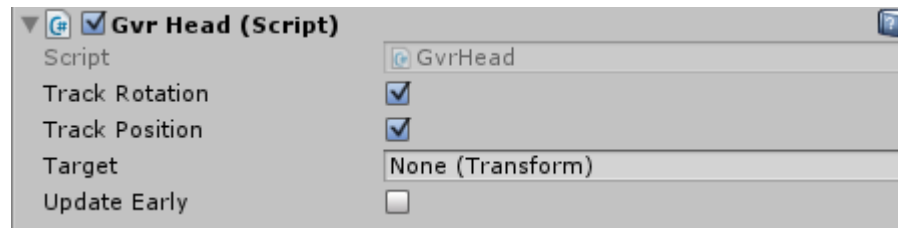


#### Where:

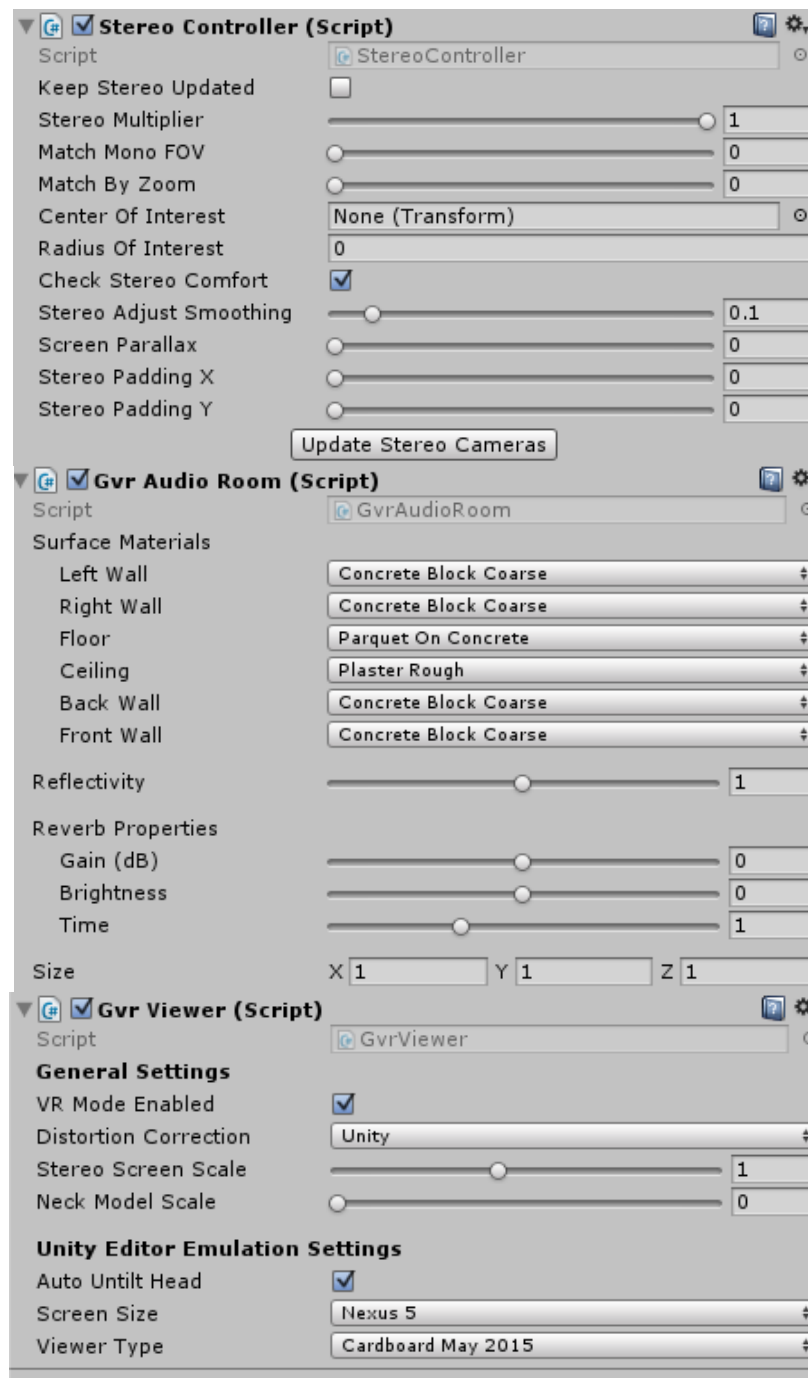
- Head is main holder as Empty object for correct pivots. Set up position of this object like on follow screenshot:



- CameraHolder as Empty object as child holds the MainCamera for correct rotations. This object need to set up on zero positions and rotations in the local coordinates. Attach follow script on this object:

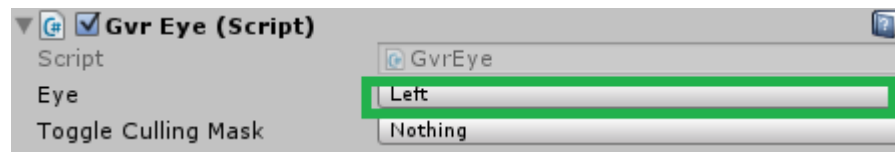


- Main Camera using for main view and need to attach follow scripts:

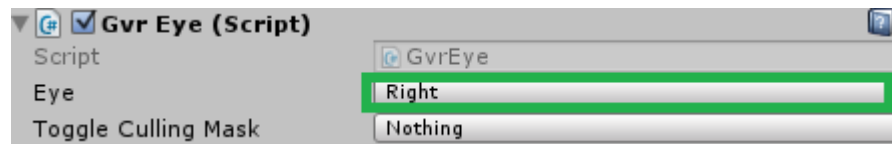


- Main Camera Left and Main Camera Right it's a Camera objects and using for drawing screen in two eyes and need to attach this script on two this cameras:

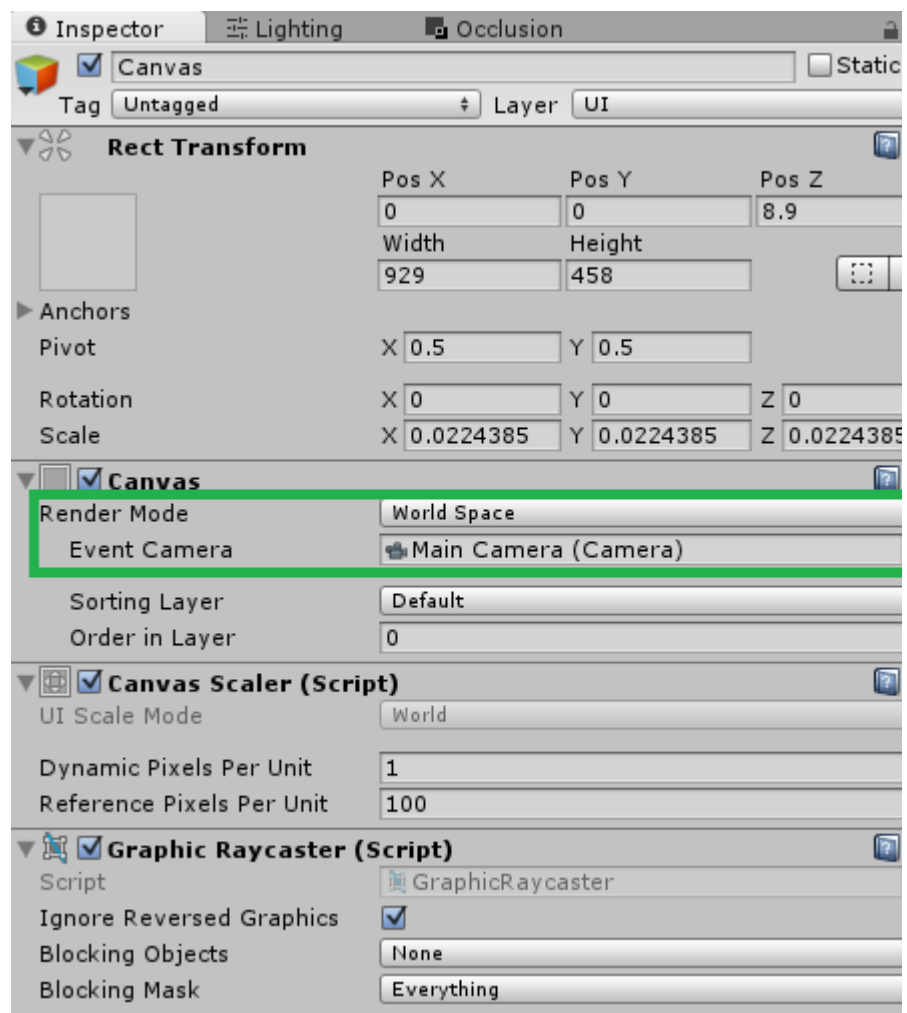
Just set Eye mode to Left option for Main Camera Left



And set Eye mode to Right option for Main Camera Right



**Note!** All Canvases need to be in world space:



3. For Gear VR you will need signing your application generating and adding into project your osig file and put it into folder:

**Assets\Plugins\Android\assets\<your osig file>**

4. Apply checkbox “Virtual Reality supported” in player settings:

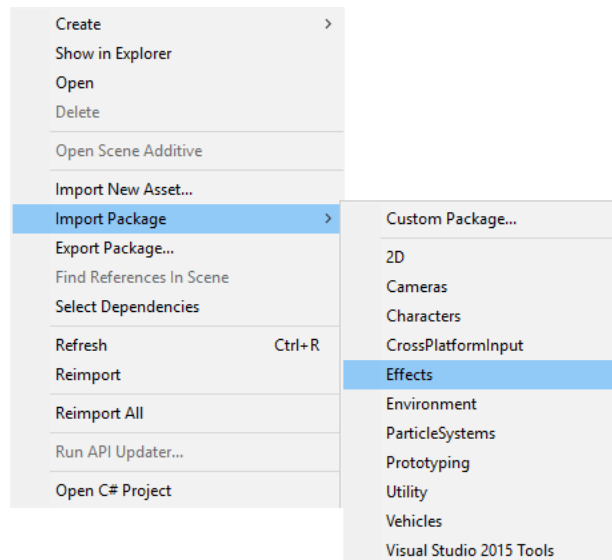


5. Turn off CameraRotation.cs script on main camera for VR mode but turning on without VR mode support for testing or using gyroscope on mobile
6. Be careful with adjustment graphics shaders for mobile VR or turning it off for best FPS increasing
7. Setting up materials:
  - Smoke material need to contain follow shader or more optimized
  - Mobile\Particles\VertexLitBlended
  - Other particles shaders need to set for mobile
  - Directive called ANDROID, WINDOWS, EDITOR indicates the desired method will be used on any platform.



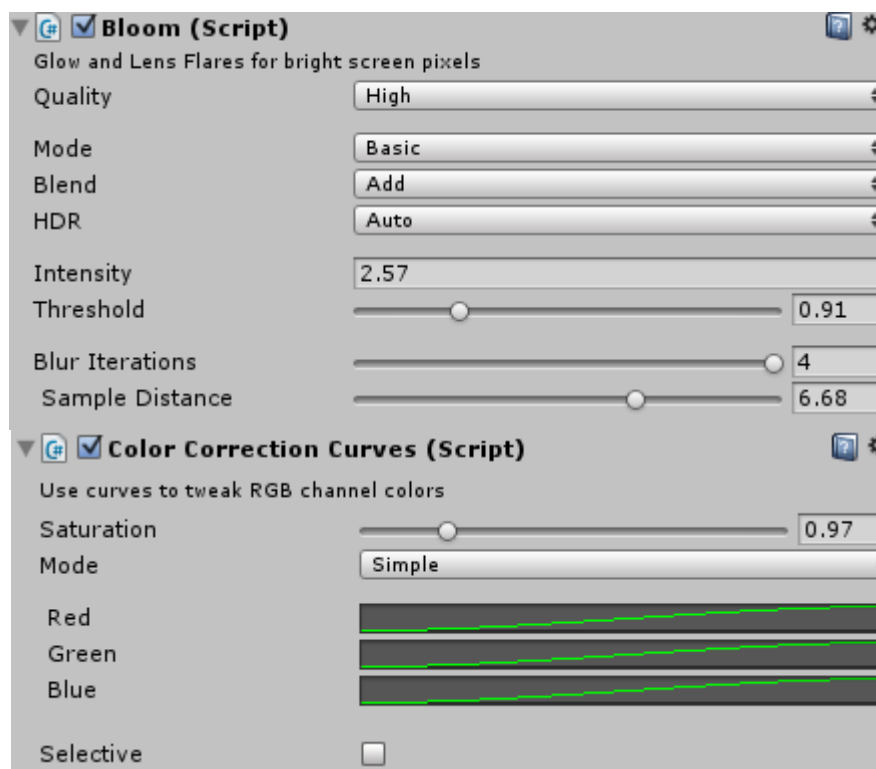
## Installation for Oculus Rift DK2 / CV1 for best graphics (Advance)

- Switch to Windows platform in the build settings
- Import Standard Post Process Effects in to your project:



## MainMenu Map Configuration

1. Attach follow post-process effects to main camera like on next screenshot:



## Scripts

### MMLogic.cs

Describes main menu functions and it's a main script for Main menu game logic.

Scenemanager it's not a **SceneManager** method using for loading levels.

Scenemanager it's gameobject (prefab) in Hierarchy window for control game modes for example player want return to main menu or restart the level and using for non destroyed sounds (include AudioSource component) or any parameters if level was re-loaded and this gameobject must be never destroying.

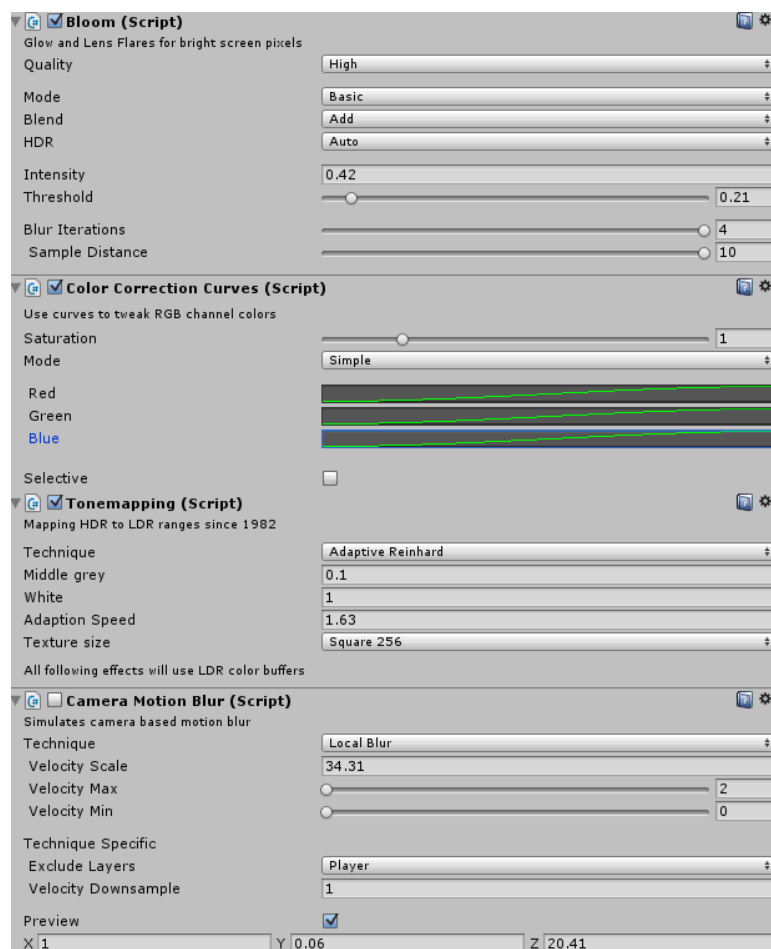
SCENEMANAGER

### SceneGM.cs

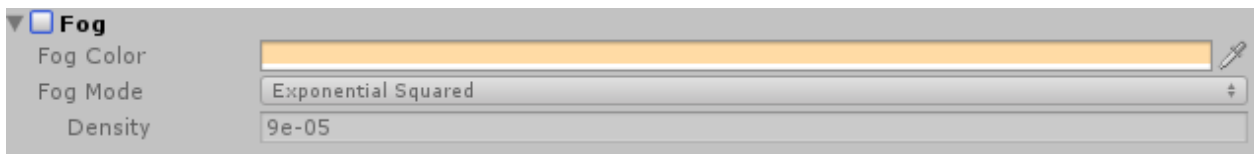
Using for Scenemanager object for exchanging data between the main scripts during gameplay.

## Level 2 Map Configuration

2. Attach follow post-process effects to main camera like on next screenshot:



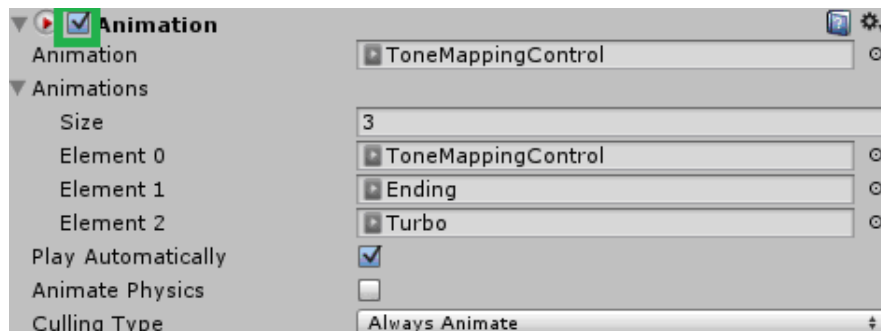
3. Set up fog color in Lighting settings and Density like next screenshot u can turn it on for more misty:



- 2.1. In Lighting setting turn off Auto baking lightmap and click on bake button for calculating only reflection probes if all objects aren't set up as static. It needs when level was reset the realtime scene may be darker.



4. Attach follow animations to Main camera in Animation component for control postprocesses and interface. Set true state in the checkbox. Samples are in the Animations folder.



5. Uncomment next lines in follow scripts:

GameLogic.cs:

On line 106  
On line 191  
On line 294  
On line 414  
On line 518  
On line 530  
On line 556

5. Turn off CameraRotation.cs script on main camera for VR mode but turning on without VR mode support.

**For now you ready to play!!! Just build for platform as you want.**

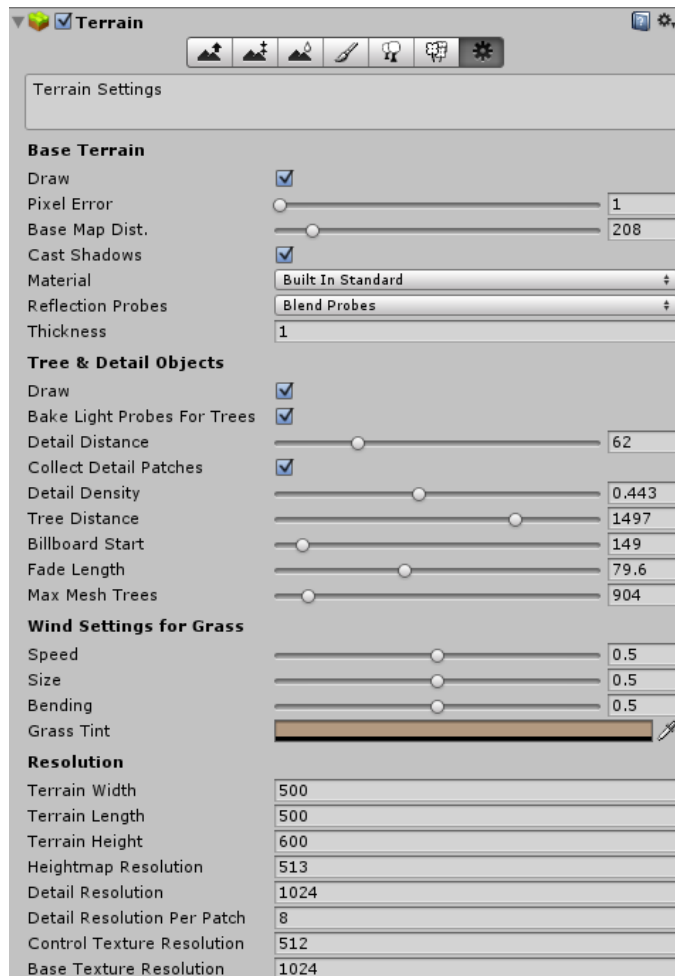
## How to play

Use VR Headset and when the game will running you need control rotations of the head.

**Additionally:**

## **Terrain**

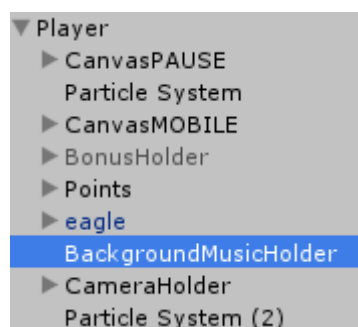
As you wish terrain can be most detailed and graphics can be drawn away. For Example you can set some parameters like on the next screenshot for PC VR mode:



Or you can use default parameters in this asset for mobile VR.

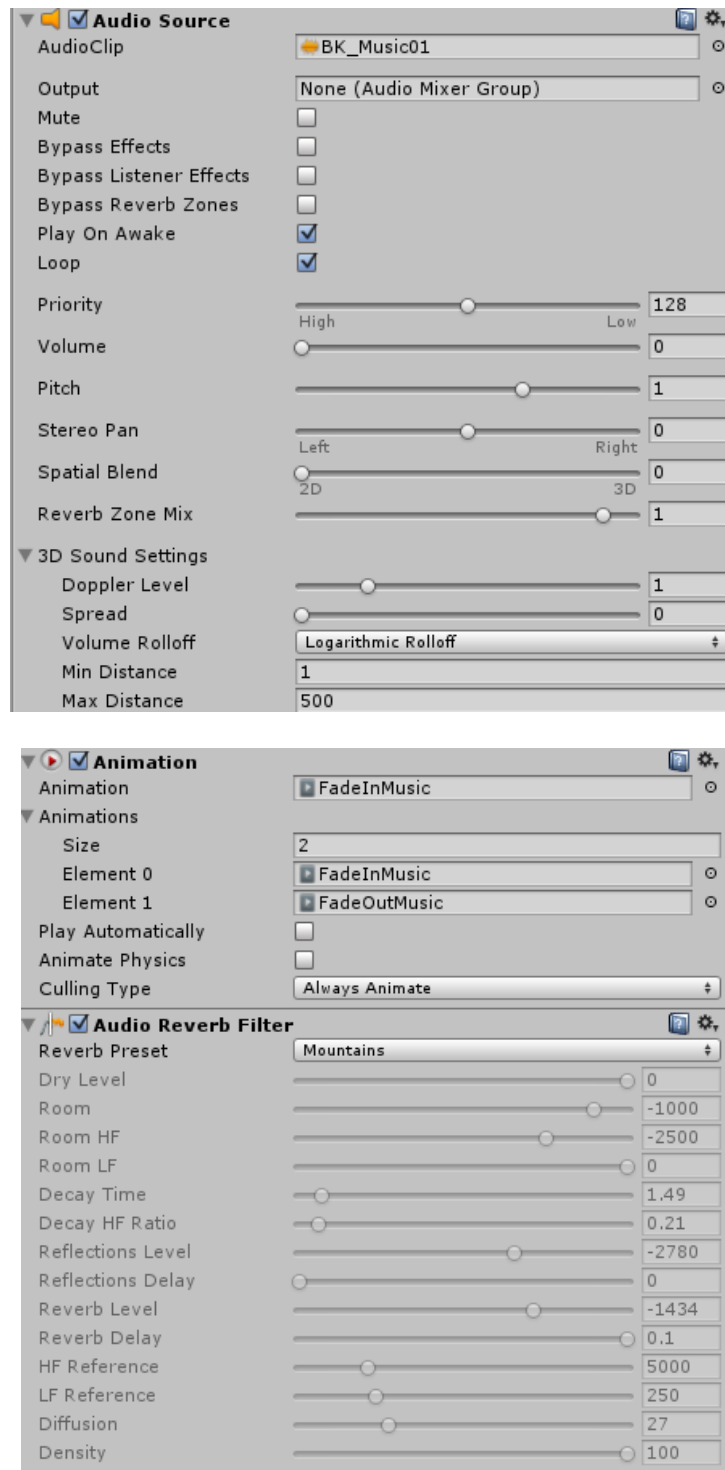
## **Background Music**

There are 3 background music in this package as an example of one track corresponds to one game mode.



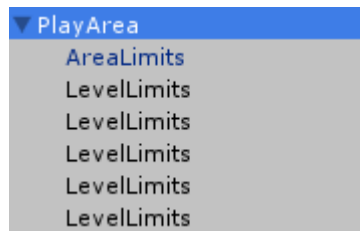


BackgroundMusicHolder must be child in Player parent object and need to attach the following components to this object AudioSource and Animation component and Sound Filter (as you want):



FadeInMusic and FadeOutMusic animations control background music volume of BackgroundMusicHolder when level or gamemode re-loaded.

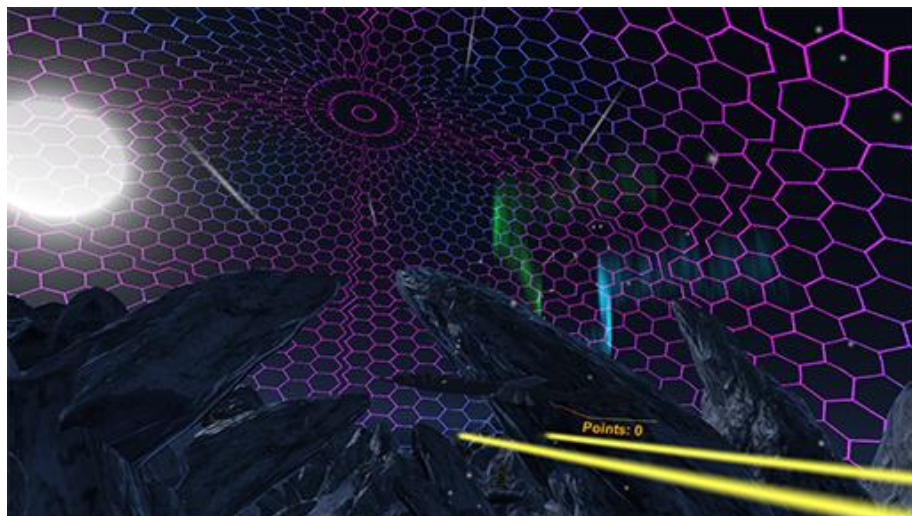
## Area limits and collisions control



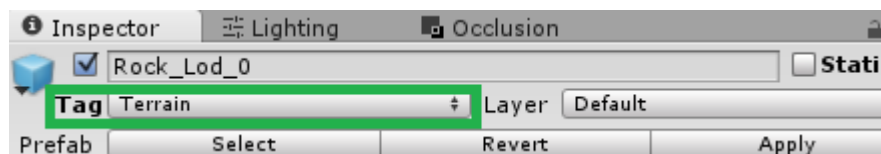
Play area in the Hierarchy window is designed for the player to control the out of the map.

Otherwise eagle dies and the game map will be reloaded with the current game mode.

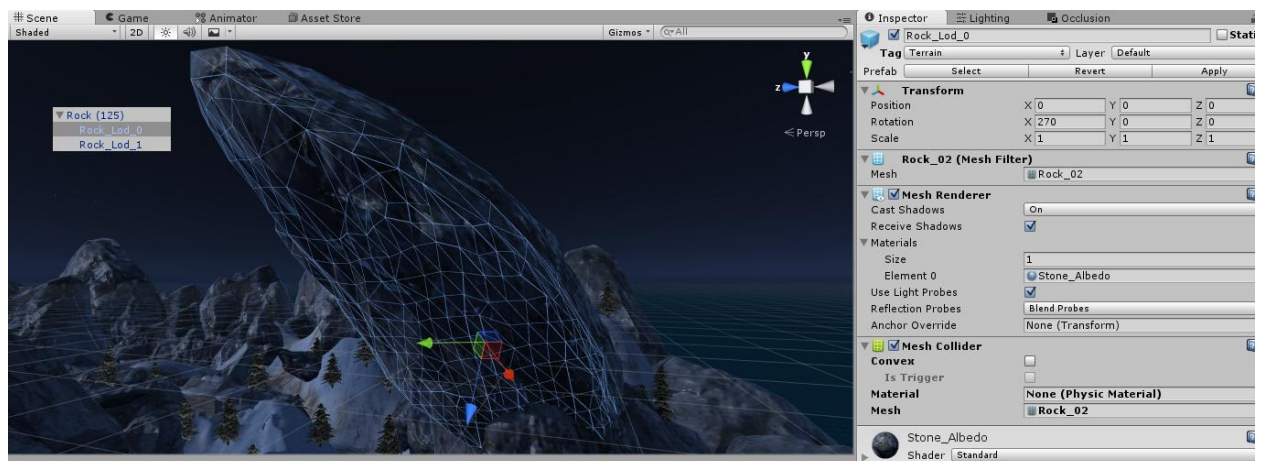
Here LevelLimits are colliders with trigger mode and AreaLimits it's graphical display in the playing area as a grid.



Any collided objects must be set Terrain tag and include collider form like a box, sphere, mesh or any other graphic collider form.



For example this rock has mesh collider and if player has collided with this rock then his eagle will die:



## Scripts

Following scripts controlling game mechanics interface and sounds.

### GameLogic.cs

It's a main script for the gameplay and placed on Main Camera. Contains 3 game modes. First game mode it's for pick up items and guide for new players. At the second game mode you will need catch the fishes. And at the third game mode you will need to finally end the game knock off rivals.

For add new game mode you need to adhere to the following structure:

```
if (gamemode == <Number of gamemode>){  
  
    //Do something  
  
}
```

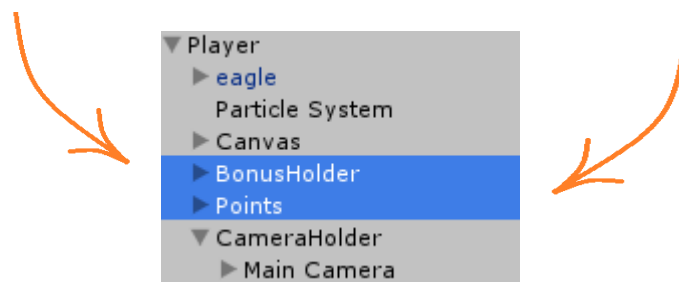
For example the 4th game mode reset player position to the next point on map and reset the rotation of main camera and showing global message about current quest like this

```
Global_Messages.text = "Get 10 fishes quickly!";
```

- Platform definition:  
For mobile used fade in\out animation of **UI panel**;  
For PC used fade animation of postprocess **Tonemapping**
- **TrailRenderers** need to be cleared before your character will be teleported to the next position
- Waiting some time set the new position of player (**SpawnZone1** in hierarchy) and screen will opened again
- Show quest message



For showing animation of bonuses or quest time used the value called `BonusHolder` and `Points` (as holder) in the hierarchy as child of the player parent holder. You can use `BonusHolder` for extended gameplay.



And finally the last game mode 5. This game mode shows chase game between 3 players for single player mode.

Other bots follow the animated point and this package show simple “fake” AI with animations.

Other bots called `EagleBotHolder_1` and `EagleBotHolder_2` in the hierarchy.

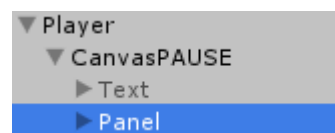
After placing player to position called `ZoneSpawn2` for this game mode show the global message for quest again and prepares eagle bots to be conquered.

Passing all the points the game ends logo.

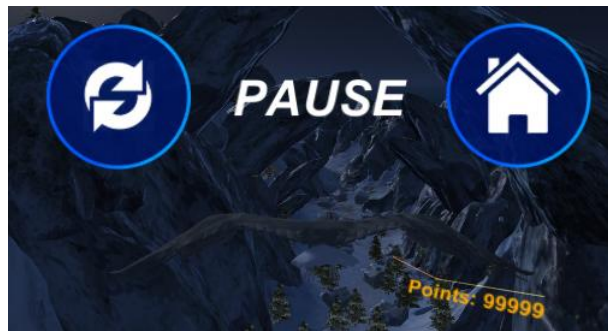
Pause Menu placed on 378 line in this script and use follow methods:

Restart and MainMenu on lines 512 and 524.

Pause Menu panel placed in CanvasPause object.



And if player clicked on Escape button or Return button on HMD like on Gear VR then pause menu will be show:



## BotSpeed.cs

Sets the speed of other eagles and randomize flying animations.

## CameraRotation.cs

Using for Main Camera rotation in the editor or windows\android platform if you will not use it for VR mode.

## ScalingObjectByDistance.cs

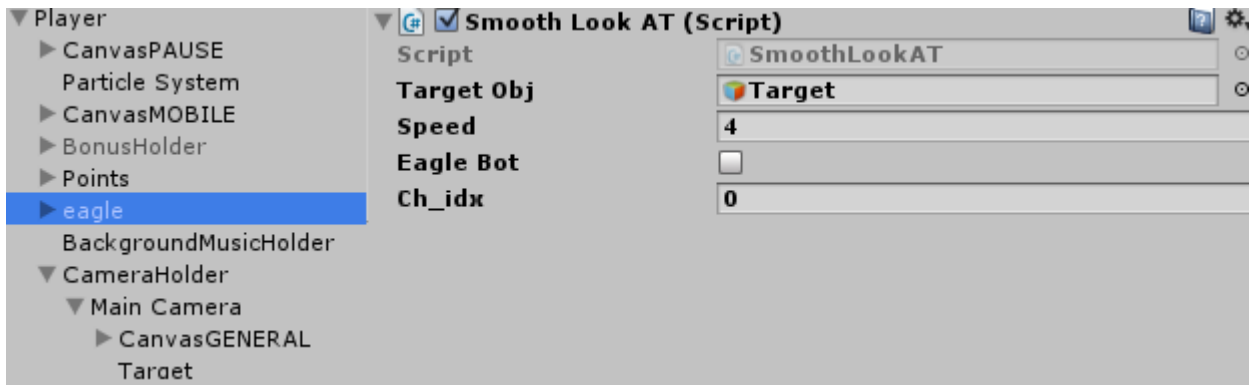
Using for Text meshes above checkpoints. For Example: Next Checkpoint will be scaled if the player is close or farther away from the checkpoint.

## SmoothLookAT.cs

```
void Start () {
    if (EagleBot)
    {
        RandomDirectionX = Random.Range(-20,5);
        RandomDirectionY = Random.Range(-20, 10);
    }
}

// Update is called once per frame
void Update () {
    if (EagleBot)
    {
        TimeDirection += Time.deltaTime;
        if (TimeDirection >= 1)
        {
            RandomDirectionX = Random.Range(-20, 15);
            RandomDirectionY = Random.Range(-20, 10);
            TimeDirection = 0;
        }
        var targetRotation = Quaternion.LookRotation(targetObj.transform.position - transform.position) * Quaternion.Euler(RandomDirectionX, RandomDirectionY, 0);
        transform.rotation = Quaternion.Slerp(transform.rotation, targetRotation, speed * Time.deltaTime);
    }
    else
    {
        var targetRotation_1 = Quaternion.LookRotation(targetObj.transform.position - transform.position);
        transform.rotation = Quaternion.Slerp(transform.rotation, targetRotation_1, speed * Time.deltaTime);
    }
}
```

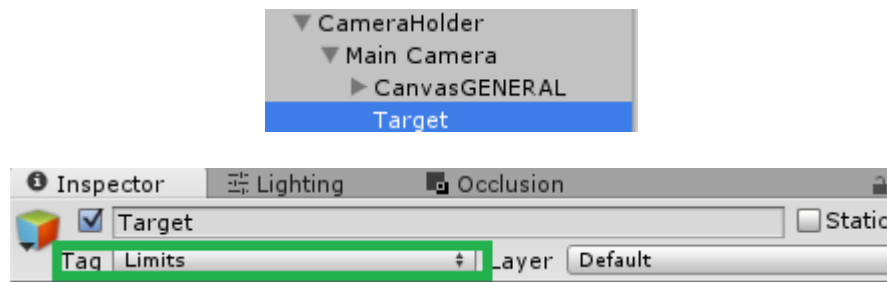
Describes the player or other players behavior. The player character will be always follow to target object placed in the hierarchy as child of the Main Camera parent  
TargetObj need attach to eagle of this script for player character



and need attach to other EagleBots in that scripts called TargetBots or any called Targets like (TargetBotsFishMode) object in the hierarchy.

### LevelLimits.cs

Use for checking if player out of map. The Target object gets information about collisions and need to be set Limits tag on it.



### LifeControl.cs

Using for checking if eagle was collided. For the future updates it can be using for contain health point of eagle and other game mechanics.

### Visibility.cs

Using for show or hide object placed in the hierarchy at realtime.

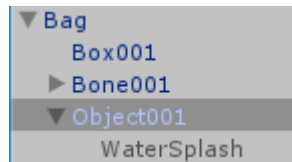
**Note!** Turn off CameraRotation.cs script on main camera for VR mode but turning on without VR mode support for testing or using gyroscope on mobile.

### Animated\_UV.cs

This script plays texture animation by UVW's on waterfalls

### BagLogic.cs

This script need to be attached to Object than will be collided with other colliders. See more comments in the script.

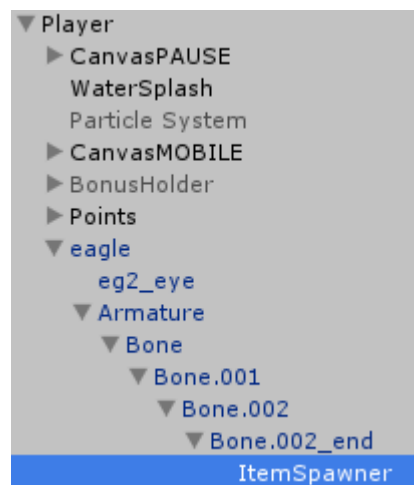


## EagleBotLogic.cs

Checks if eagle bots will fighting.

## EagleItemSpawner.cs

Describes eagle player shooting. You need to attach this script to item spawner gameobject like this:



## EaglesZone.cs

It's a collider in the hierarchy called EaglesZone with this script and checks if eagle bots inside then they got one fish.

## FishLogic.cs

This script must be using between fish zone spawn in hierarchy window and fish prefab.  
Describes behaviour of fishes.



FishZoneSpawn rotates FishView as target for fishes for they follow.

## PickItem.cs

It located in areas where will be made the selection of a specific item gameobject for example Dead\_plants gameobjects in hierarchy.

When an object is received the value of this item is sent to EagleItemSpawner script.



## Frequently Asked Questions

For better understanding see comments in scripts.

**Q.** Why this asset doesn't contain Image Effects package?

**A.** The reason for the lack of standard assets package:  
Due to possible compatibility issues between Unity versions.

**Q.** Why this asset doesn't contain Google Cardboard package?

**A.** Because this package is constantly updated and its implementation can not conform to the description in this documentation in case you need to refer to the official documentation for the implementation of Google cardboard components in your project.

Maybe you interesting more assets for virtual reality by Sergey Shushunov in the Asset Store:

