

FIRST PERSON VIEW

MANUAL

Index

[Introduction](#)

[Features](#)

[Game Features](#)

[Script Features](#)

[Editor Features](#)

[Limitations](#)

[Performance](#)

[How it Works](#)

[Layers](#)

[Camera](#)

[MainCamera](#)

[First Person View Camera](#)

[Image Effects Camera](#)

[First Person Object and First Person View](#)

[Setup](#)

[Scripting Functionalities](#)

[FPV_Container.cs](#)

[IFPV_Object.cs](#)

[DEMO](#)

Introduction

First Person View is an asset that allows you to have a First Person Perspective where first person objects do not clip through the environment, have a separate field of view from the environment and can receive shadows from the environment.

In this document i'll describe how this method works and how to make it work in your project.

Every script is well documented and well structured for you to be able to understand how everything works and be able to modify and improve it for your specific needs.

This asset was made for free for everyone to support the Unity Community.

Features

Game Features

- First Person View models don't clip through the environment
- First Person View models receive shadows from the environment
- Independent Field-of-View between First Person View and the World View
- Ability to choose objects that will or not cast shadows on the First Person View models

Script Features

- Ability to automatically change between World View + First Person View and World View only. (useful when the player interacts with the environment)
- Ability to assign and remove objects to/from the First Person View perspective
- Automatic Layer assignment for First Person Objects. It preserves original layers when changing the objects back to World View
- Transform a point and/or direction from First Person View to World View

Editor Features

- Automatically create the Camera asset with First Person View ready
- Add/remove First Person Object component to/from selected objects

Limitations

As of now, the limitations of this system are:

- No support for Unity's Terrain Engine and Terrain Trees. To be able to use the terrain with this system, you will need to put the terrain in another layer that is not rendered by the First Person View Camera. Trees outside of the terrain system will work good.
- This system only works in Deferred Rendering and Legacy Deferred. It does not work in Forward Rendering.

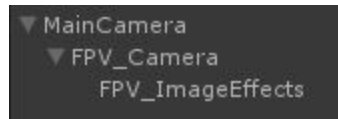
How it Works

Layers

For this setup to work, you need to set a layer named “FirstPersonView”. A script will automatically find its id, so no need to set the layer in any script.

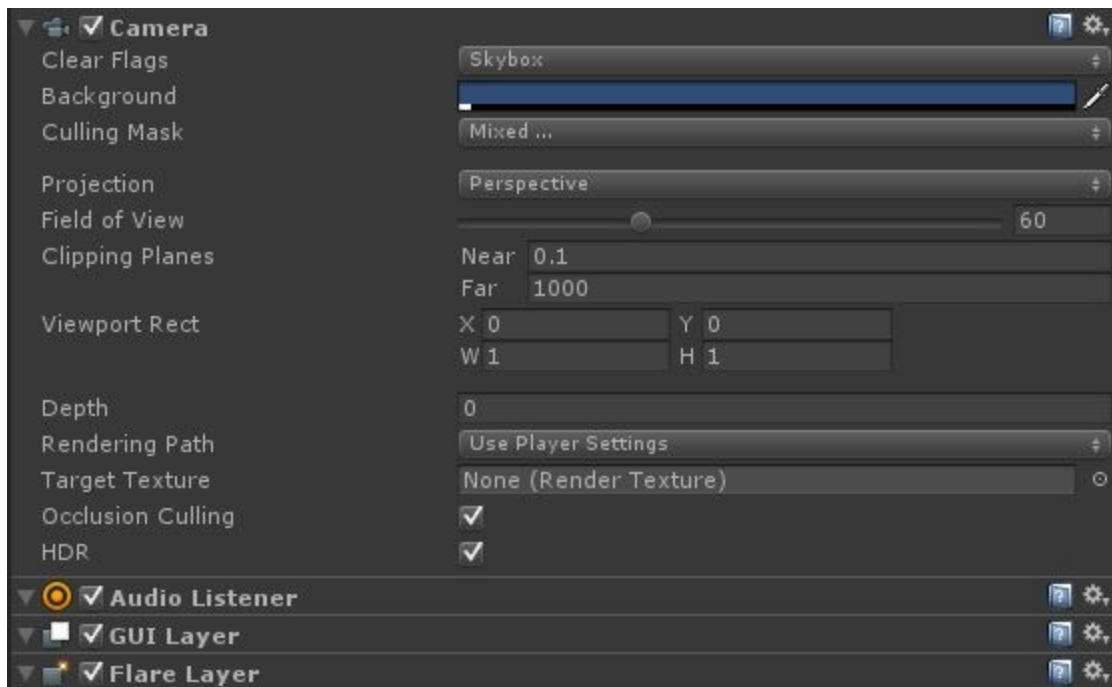
Camera

The Camera object is a 3 camera



setup

MainCamera



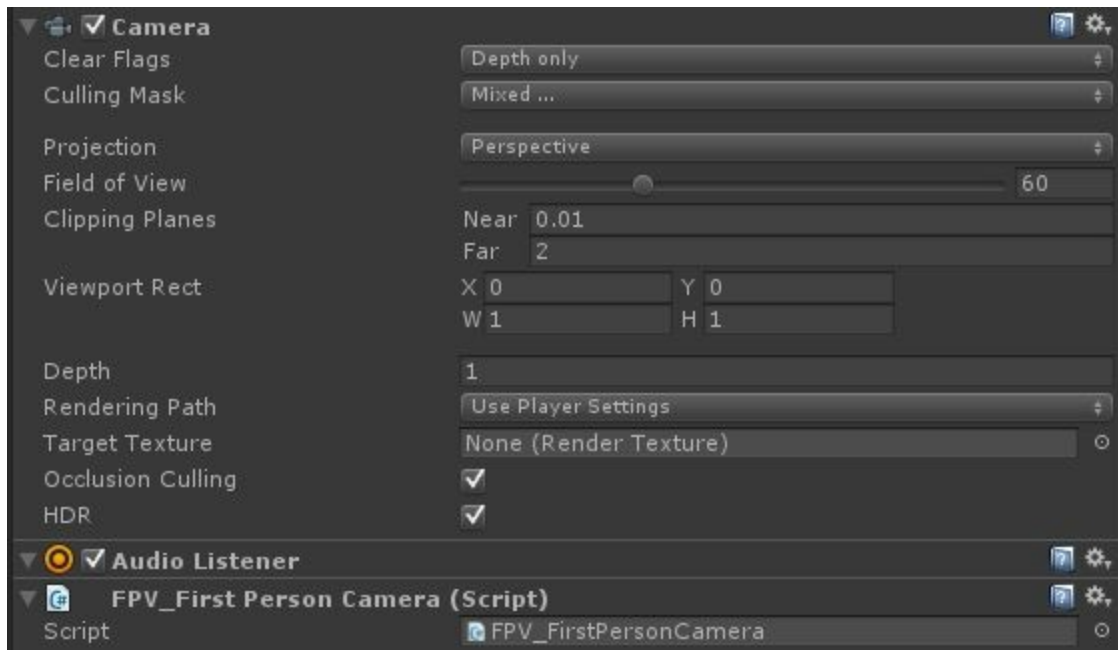
The main camera is the camera that renders the environment. This camera should have the FPV_WorldCamera attached to it.

Set the far and near clipping planes to your specific needs.

All Depth Based Post Processing Effects should be used in this camera. It can also be used in the last camera, but there are limitations due to how the depth buffer is modified. Depth based effects won't affect the FPV models through this camera.

When this camera is rendering, it will set a flag telling the system that this camera is going to render.

First Person View Camera



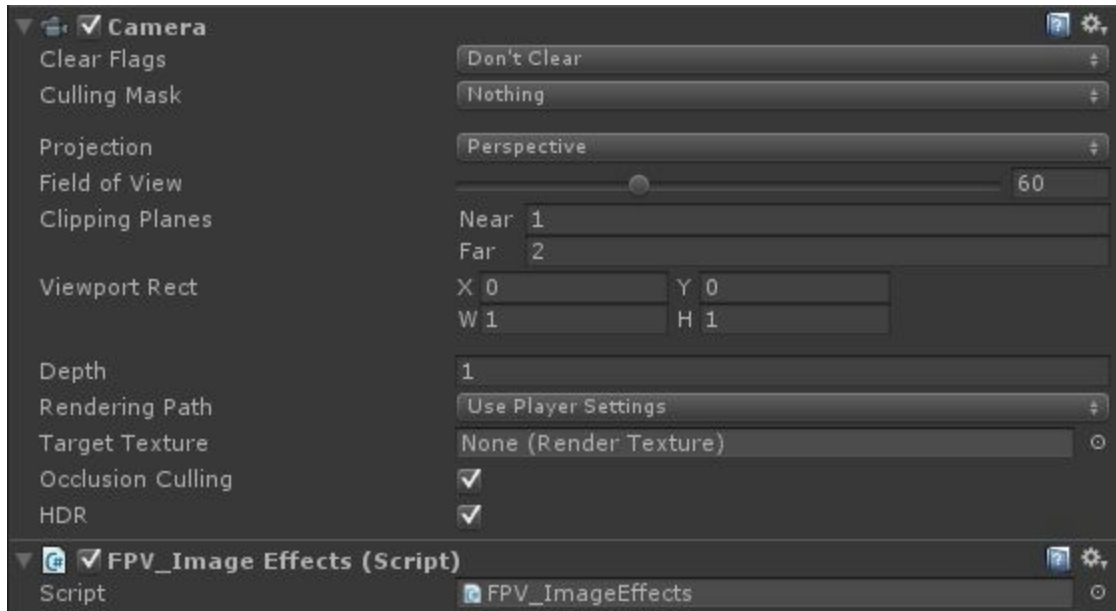
The First Person View Camera will render the first person perspective.

How this camera is setup, the camera will only clear the depth buffer and it will render every layer (except layers you know the player will never be close to, like clouds, far buildings, etc). For this camera, you need to set the Clipping Planes to small values, like [0.01, 2]. You can increase it if really necessary, but don't increase it too much.

The `FPV_FirstPersonCamera` component is attached to this camera, is what will handle the First Person View.

When this camera is rendering, it will set a flag telling the rest of the system that this camera is going to render next. This is used for `OnWillRenderObject`.

Image Effects Camera



The Image Effects Camera is set in a way that will not render anything.

The `FPV_ImageEffects` component will use the `WorldCamera` and the `FPV_Camera` and make them render into a custom `RenderTexture` that will then be used by this camera.

Due to how Unity works, this makes it possible to use the depth buffer from all cameras in a simple way.

All effects that are not Depth Based should be used after this component.

Some Depth Based Image Effects will work here, but some are better used in the `WorldCamera`.

First Person Object and First Person View

First Person Objects are objects that will have a special component attached to them that will allow the system to enable shadowsOnly/Disable/Enable with ease.

When the FPV_Object component is attached to a gameobject, it will automatically create FPV_Renderer components for all objects inside it that have a renderer component.

There are two types of FPV_Objects, one is going to change the shadow mode of the renderer component, and the other is going to disable the renderer.

Due to this, these two components need to be disabled/enabled in different places at different times.

To simplify, here's the process in a sequential way:

On Update

- FPV_DisableOnly will check if a renderer inside was changed by the FPV. If it was, it means that the renderers were disabled for the First Person View, so in this Update, the method will re-enable the renderers so they are active when the World Camera is going to render.

World Camera - OnWillRenderObject

- FPV_Renderer will first check if it is not a first person object. Since the World Camera is currently rendering, it will check if the renderer was changed by the First Person Camera. If it was, it will revert it back to normal so the World camera will render it correctly
- FPV_Renderer_DisableOnly will first check if it is not a first person object. It will then check if the world camera is rendering and will enable the First Person View for this renderer to prepare it for the First Person Camera.

First Person Camera - OnWillRenderObject

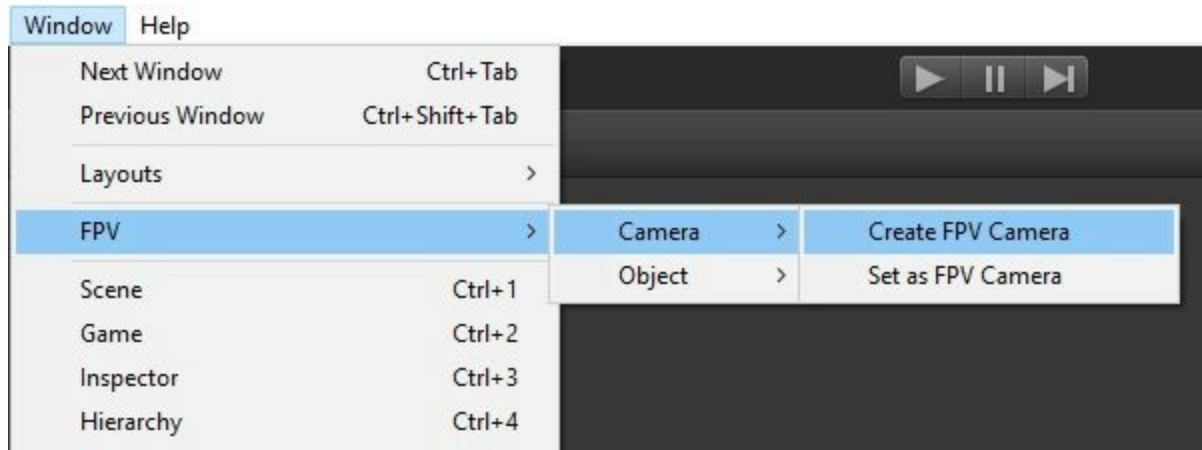
- FPV_Renderer will first check if it is not a first person object. Since the First Person Camera is rendering, it will enable the First Person View of the renderer (in this case, it will set the shadow casting mode to ShadowsOnly.

Setup

First, add a new Layer named "FirstPersonView".

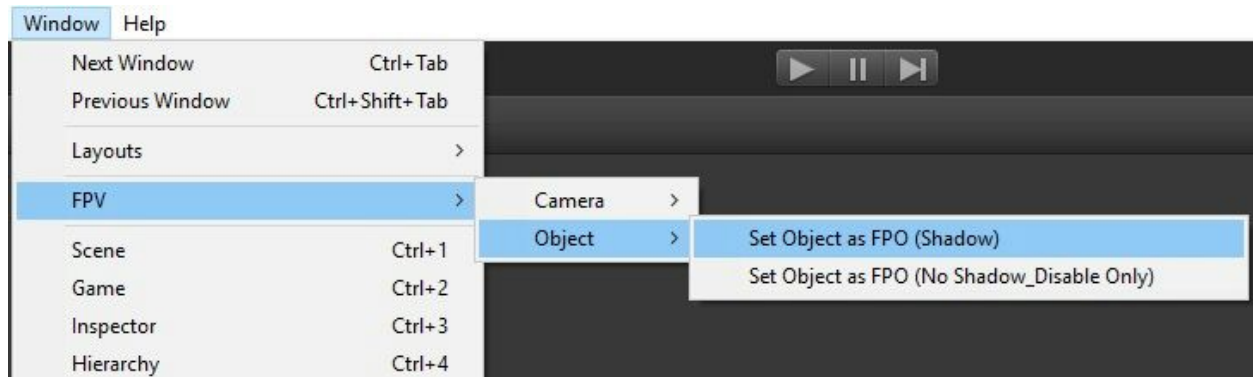
Set the rendering path to Deferred in the player options.

To setup this system, first create a new Camera Object through Window/FPV/Camera/Create FPV Camera



This will automatically create all cameras and setup every component and their properties.
After this, your camera is ready to be used.

Next, for every object you have at the top of its hierarchy, add the FPV_Object component to it through Window/FPV/Object/



Select the option you want for all the renderers inside your object.

And you are all set.

Scripting Functionalities

FPV.cs

- Access to all cameras of this system
- Contains the FirstPersonRenderLayer
- `public static Vector3 FPVPointToWorldPoint(Vector3 point)`

Transforms a point from World -> FPV screen -> World

- `public static void FPVToWorld(Vector3 point, Vector3 direction, out Vector3 resPoint, out Vector3 resDirection)`

It will transform a point and direction from World -> FPV Screen -> World

- `public static void FPVToWorld(Transform trans, out Vector3 resPoint, out Vector3 resDirection)`

It will transform a point and forward direction from a Transform from World -> FPV Screen -> World

IFPV_Object.cs

- `public void SetAsFirstPersonObject()`

Call this method when you want an object that has this component to be transformed into a First Person View object. It will automatically handle all renderers inside it (except for other IFPV_Objects inside it. Those you also need to do it manually)

- `public void RemoveAsFirstPersonObject()`

Call this method when you want an object that has this component to be transformed into a World View object. It will automatically handle all renderers inside it (except for other IFPV_Objects inside it. Those you also need to do it manually)

- `public bool IsFirstPersonObject()`

Checks if this IFPV_Object is a First Person View object.

DEMO

An example scene is included in the project that will simply show how everything is working together.

I used the First Person Controller by Unity. I put it in another package in case you don't want to import it into your project.

Controls:

- normal fps controls for movement
- 'P' will spawn a 'weapon'
- 'i' will attach/detach the weapon from the player. This is used to demonstrate how you can translate objects from world to first person and vice-versa with ease.
- ',' and '.' will increase/decrease the World's Field of View
- 'n' and 'm' will increase/decrease the First Person View Field of View