

# Documentación Sistema de Cámaras

## Sistema de Cámaras

### Índice

Este es un trabajo para Unity 3D, pensado como un sistema de cámaras que puede ser integrado y modificado según las cámaras que necesite el usuario del usuario.

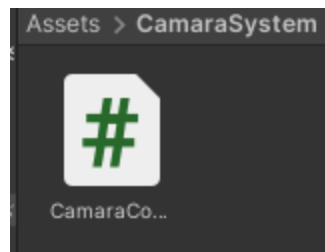
Su script contiene una lista de cámaras permitiendo su adaptabilidad y/o experimentación de las mismas.

### Instalación

1- Acceder al siguiente link para descargar el package de archivos de forma rápida:

[Sistema-De-Camara--PV2/CamaraSysFresh.v1.unitypackage at main · Sp0my/Sistema-De-Camara--PV2 \(github.com\)](#).

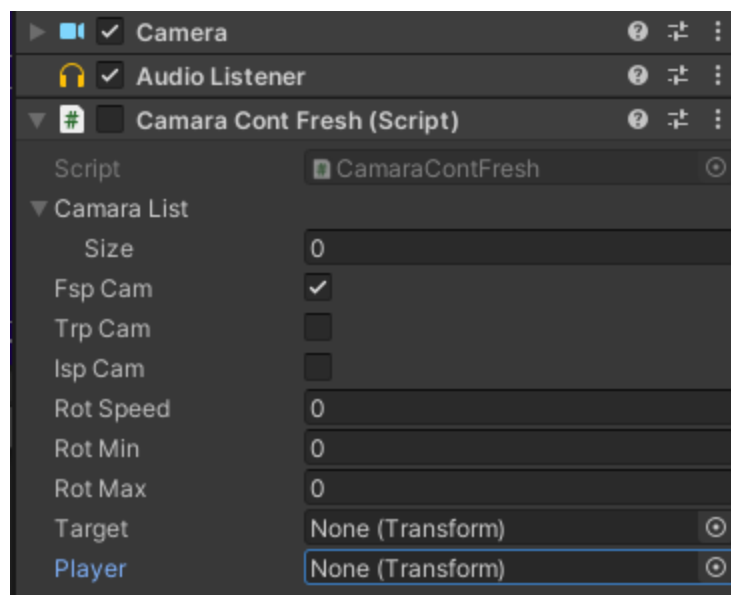
2- Al descargar el package, deberás buscar una carpeta con el nombre **CamaraSystem**, la cual contendrá dentro el script **CamaraControllerFresh**:



# Contenido

## ▼ CamaraControllerFresh

Deberás asignar el script al objeto de tu personaje y, al hacerlo, en el inspector aparecerán los siguientes apartados:



**Size** te permitirá definir el número de cámaras en tú lista.

**Rot Speed** manejará la velocidad a la que se mueve tú cámara.

**Rot Min** y **Rot Max** controlarán los ángulos de rotación.

**Target** y **Player** deberán ser asignados para que puedan ser identificados dentro del script, siendo target un objeto vacío, hijo del objeto jugador.

## Configuración

Dentro de este script encontraras la variable que controla cuantas cámaras máximas podrá recibir tu lista y las variables booleanas que activan y desactivan las mismas. Estas ya vienen con 3 tipos de cámara (Primera Persona, Tercera Persona e Isométrica), pero se pueden eliminar o agregar cámaras, comenzando su modificación en estas variables.

```
public GameObject[] camaraList;  
int ncamaras = 3;  
public bool fspCam = true;  
public bool trpCam = false;  
public bool ispCam = false;
```

En **Update** encontrarás primero el ciclo que activa las cámaras, teniendo que modificar este en caso de que se eliminen u agreguen cámaras en la lista y variables.

```

void Update()
{
    //ACTIVADOR DE CAMARAS
    if(Input.GetKey(KeyCode.Alpha1))
    {
        Debug.Log("Primera Persona activada");
        TurnOffCams();
        camaraList[0].gameObject.SetActive(true);

        fspCam = true;
        trpCam = false;
        ispCam = false;
    }
    if (Input.GetKey(KeyCode.Alpha2))
    {
        Debug.Log("Tercera Persona activada");
        TurnOffCams();
        camaraList[1].gameObject.SetActive(true);

        fspCam = false;
        trpCam = true;
        ispCam = false;
    }
    if (Input.GetKey(KeyCode.Alpha3))
    {
        Debug.Log("Vista Isométrica activada");
        TurnOffCams();
        camaraList[2].gameObject.SetActive(true);

        fspCam = false;
        trpCam = false;
        ispCam = true;
    }
}

```

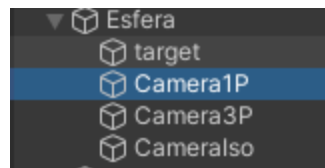
Seguido de esto encontrarás el ciclo que activa el movimiento de la cámara en los casos de Primera y Tercera Persona, de nuevo debiendo modificarlo de borrar u agregar cámaras.

```
//ACTIVACION DE FUNCIONES
if ( fspCam == true)
{
    mouseX += rotSpeed * Input.GetAxis("Mouse X");
    mouseY -= rotSpeed * Input.GetAxis("Mouse Y");
    mouseY = Mathf.Clamp(mouseY, rotMin, rotMax);

    target.rotation = UnityEngine.Quaternion.Euler(mouseY, mouseX, 0.0f);
    player.rotation = UnityEngine.Quaternion.Euler(0.0f, mouseX, 0.0f);
}
if ( trpCam == true)
{
    mouseX += rotSpeed * Input.GetAxis("Mouse X");
    mouseY -= rotSpeed * Input.GetAxis("Mouse Y");
    mouseY = Mathf.Clamp(mouseY, rotMin, rotMax);

    target.rotation = UnityEngine.Quaternion.Euler(mouseY, mouseX, 0.0f);
    player.rotation = UnityEngine.Quaternion.Euler(0.0f, mouseX, 0.0f);
}
if ( ispCam == true)
{
}
}
```

Finalmente, en el Hierarchy deberás crear las cámaras que quieras utilizar como hijos del jugador, colocando cada una en la posición desde la que desees observen al jugador y asignarlas de vuelta en la lista de cámaras





¡Y listo! Así de fácil podrás tener tu sistema de cámaras montado para tu proyecto.