using System;

using UnityEngine;

using UnityStandardAssets.CrossPlatformInput;

namespace UnityStandardAssets.Characters.FirstPerson

{

[Serializable]

public class MouseLook

{

public float XSensitivity = 2f;

public float YSensitivity = 2f;

public bool clampVerticalRotation = true;

public float MinimumX = -90F;

public float MaximumX = 90F;

public bool smooth;

public float smoothTime = 5f;

private Quaternion m\_CharacterTargetRot;

private Quaternion m\_CameraTargetRot;

public void Init(Transform character, Transform camera)

{

m\_CharacterTargetRot = character.localRotation;

m\_CameraTargetRot = camera.localRotation;

}

public void LookRotation(Transform character, Transform camera)

{

float yRot = CrossPlatformInputManager.GetAxis("Mouse X") \* XSensitivity;

float xRot = CrossPlatformInputManager.GetAxis("Mouse Y") \* YSensitivity;

m\_CharacterTargetRot \*= Quaternion.Euler (0f, yRot, 0f);

m\_CameraTargetRot \*= Quaternion.Euler (-xRot, 0f, 0f);

if(clampVerticalRotation)

m\_CameraTargetRot = ClampRotationAroundXAxis (m\_CameraTargetRot);

if(smooth)

{

character.localRotation = Quaternion.Slerp (character.localRotation, m\_CharacterTargetRot,

smoothTime \* Time.deltaTime);

camera.localRotation = Quaternion.Slerp (camera.localRotation, m\_CameraTargetRot,

smoothTime \* Time.deltaTime);

}

else

{

character.localRotation = m\_CharacterTargetRot;

camera.localRotation = m\_CameraTargetRot;

}

}

Quaternion ClampRotationAroundXAxis(Quaternion q)

{

q.x /= q.w;

q.y /= q.w;

q.z /= q.w;

q.w = 1.0f;

float angleX = 2.0f \* Mathf.Rad2Deg \* Mathf.Atan (q.x);

angleX = Mathf.Clamp (angleX, MinimumX, MaximumX);

q.x = Mathf.Tan (0.5f \* Mathf.Deg2Rad \* angleX);

return q;

}

}

}