using System;

using UnityEngine;

using UnityStandardAssets.CrossPlatformInput;

namespace UnityStandardAssets.Characters.ThirdPerson

{

[RequireComponent(typeof (ThirdPersonCharacter))]

public class ThirdPersonUserControl : MonoBehaviour

{

private ThirdPersonCharacter m\_Character; // A reference to the ThirdPersonCharacter on the object

private Transform m\_Cam; // A reference to the main camera in the scenes transform

private Vector3 m\_CamForward; // The current forward direction of the camera

private Vector3 m\_Move;

private bool m\_Jump; // the world-relative desired move direction, calculated from the camForward and user input.

private void Start()

{

// get the transform of the main camera

if (Camera.main != null)

{

m\_Cam = Camera.main.transform;

}

else

{

Debug.LogWarning(

"Warning: no main camera found. Third person character needs a Camera tagged \"MainCamera\", for camera-relative controls.");

// we use self-relative controls in this case, which probably isn't what the user wants, but hey, we warned them!

}

// get the third person character ( this should never be null due to require component )

m\_Character = GetComponent<ThirdPersonCharacter>();

}

private void Update()

{

if (!m\_Jump)

{

m\_Jump = CrossPlatformInputManager.GetButtonDown("Jump");

}

}

// Fixed update is called in sync with physics

private void FixedUpdate()

{

// read inputs

float h = CrossPlatformInputManager.GetAxis("Horizontal");

float v = CrossPlatformInputManager.GetAxis("Vertical");

bool crouch = Input.GetKey(KeyCode.C);

// calculate move direction to pass to character

if (m\_Cam != null)

{

// calculate camera relative direction to move:

m\_CamForward = Vector3.Scale(m\_Cam.forward, new Vector3(1, 0, 1)).normalized;

m\_Move = v\*m\_CamForward + h\*m\_Cam.right;

}

else

{

// we use world-relative directions in the case of no main camera

m\_Move = v\*Vector3.forward + h\*Vector3.right;

}

#if !MOBILE\_INPUT

// walk speed multiplier

if (Input.GetKey(KeyCode.LeftShift)) m\_Move \*= 0.5f;

#endif

// pass all parameters to the character control script

m\_Character.Move(m\_Move, crouch, m\_Jump);

m\_Jump = false;

}

}

}