using System;

using UnityEngine;

namespace UnityStandardAssets.\_2D

{

public class CameraFollow : MonoBehaviour

{

public float xMargin = 1f; // Distance in the x axis the player can move before the camera follows.

public float yMargin = 1f; // Distance in the y axis the player can move before the camera follows.

public float xSmooth = 8f; // How smoothly the camera catches up with it's target movement in the x axis.

public float ySmooth = 8f; // How smoothly the camera catches up with it's target movement in the y axis.

public Vector2 maxXAndY; // The maximum x and y coordinates the camera can have.

public Vector2 minXAndY; // The minimum x and y coordinates the camera can have.

private Transform m\_Player; // Reference to the player's transform.

private void Awake()

{

// Setting up the reference.

m\_Player = GameObject.FindGameObjectWithTag("Player").transform;

}

private bool CheckXMargin()

{

// Returns true if the distance between the camera and the player in the x axis is greater than the x margin.

return Mathf.Abs(transform.position.x - m\_Player.position.x) > xMargin;

}

private bool CheckYMargin()

{

// Returns true if the distance between the camera and the player in the y axis is greater than the y margin.

return Mathf.Abs(transform.position.y - m\_Player.position.y) > yMargin;

}

private void Update()

{

TrackPlayer();

}

private void TrackPlayer()

{

// By default the target x and y coordinates of the camera are it's current x and y coordinates.

float targetX = transform.position.x;

float targetY = transform.position.y;

// If the player has moved beyond the x margin...

if (CheckXMargin())

{

// ... the target x coordinate should be a Lerp between the camera's current x position and the player's current x position.

targetX = Mathf.Lerp(transform.position.x, m\_Player.position.x, xSmooth\*Time.deltaTime);

}

// If the player has moved beyond the y margin...

if (CheckYMargin())

{

// ... the target y coordinate should be a Lerp between the camera's current y position and the player's current y position.

targetY = Mathf.Lerp(transform.position.y, m\_Player.position.y, ySmooth\*Time.deltaTime);

}

// The target x and y coordinates should not be larger than the maximum or smaller than the minimum.

targetX = Mathf.Clamp(targetX, minXAndY.x, maxXAndY.x);

targetY = Mathf.Clamp(targetY, minXAndY.y, maxXAndY.y);

// Set the camera's position to the target position with the same z component.

transform.position = new Vector3(targetX, targetY, transform.position.z);

}

}

}