GAME OBJECT

// adding code to play the sound

~~Using FMODUnity;~~

Using FMOD.Studio; // for the instance

Using FMODUnity;

//this is an annotation ??? so Unity will complain it is missing cfr. multiple coins

[RequireComponent(typeof(StudioEventEmitter))]

public class Coin : MonoBehaviour

{

~~[SerializeField] private EventReference coinCollectedSou~~nd // this is an eventreference type variable called coinCollectedSound

Private EventInstance playerFootstepsSound; //declaring a private event instance variable for the footstep instance

Private StudioEventEmitter emitter;

// this is the method we will use to play the sound where we will call the audio manager

private void CollectCoin()

{

~~AudioManager.instance.PlayOneshot (coinCollectedSound, this.transform.position);~~

AudioManager.instance.PlayOneShot(FMODEvents.instance.coinCollectedSound, this.transform.position);

Emitter.Stop();

}

// start method for the footsteps to initialize the footsteps sound/event, because you always have to start, stop and release an event-instance

Private void Start()

{

playerFootstepsSound = audiomanager.instance.CreateEventInstance(FMODEvents.instance.playerFootstepsSound,

Private void Start()

{

emitter = AudioManager.instance.InitializeEventEmitter(FMODEvents.instance.coinIdleSound, this.gameObject)

emitter.Play();

}

);

}

// this is for the footstep eventinstance

Private void FixedUpdate()

{

Inside this fixed update you’ll find the function that makes the object move

UpdateSound();

}

// new method for the footsteps, to see if we are moving or not

Private void UpdateSound()

{

if ( moving and grounded)

{

PLAYBACK\_STATE playbackState; // declaring a playback state

playerFootstepsSound.getPlaybackState (out playbackState);

if (playbackState.Equals (PLAYBAK\_STATE.STOPPED))

{

playerFootstepsSound.start();

}

}

else

{

playerFootstepsSound.stop(STOP\_MODE.ALLOWFADEOUT);

}

}

}