FOOTSTEP TERRAIN ANIMATION ; FAMA

**C# code for playing footsteps events in Unity**

After the preparations we select the GameObject in which the **Animator** component responsible for the player animations is located. There we create a script for the footsteps sounds. We first declare an enumerator for the different ground types and of course also the FMOD Event Instance:

private enum CURRENT\_TERRAIN { GRASS, GRAVEL, WOOD\_FLOOR, WATER };

[SerializeField]

private CURRENT\_TERRAIN currentTerrain;

private FMOD.Studio.EventInstance foosteps;

We then create the DetermineTerrain method, which will help us to identify he correct ground type the player is currently walking on:

private void DetermineTerrain()

{

RaycastHit[] hit;

hit = Physics.RaycastAll(transform.position, Vector3.down, 10.0f);

foreach (RaycastHit rayhit in hit)

{

if (rayhit.transform.gameObject.layer == LayerMask.NameToLayer("Gravel"))

{

currentTerrain = CURRENT\_TERRAIN.GRAVEL;

}

else if (rayhit.transform.gameObject.layer == LayerMask.NameToLayer("Wood"))

{

currentTerrain = CURRENT\_TERRAIN.WOOD\_FLOOR;

}

else if (rayhit.transform.gameObject.layer == LayerMask.NameToLayer("Grass"))

{

currentTerrain = CURRENT\_TERRAIN.GRASS;

}

else if (rayhit.transform.gameObject.layer == LayerMask.NameToLayer("Water"))

{

currentTerrain = CURRENT\_TERRAIN.WATER;

}

}

}

Basically we send a ray from the player position to Vector3.down with a distance of 10 game units. In the if statements we check if the ray hits a GameObject with the respective layer. If so, we change currentTerrain. We put this method into Unity’s Update() method.

Now we’ll take care of playing the footstep sounds. We create a very simple PlayFootsteps() method:

private void PlayFootstep(int terrain)

{

foosteps = FMODUnity.RuntimeManager.CreateInstance("event:/Footsteps");

foosteps.setParameterByName("Terrain", terrain);

foosteps.set3DAttributes(FMODUnity.RuntimeUtils.To3DAttributes(gameObject));

foosteps.start();

foosteps.release();

}

The method takes an int argument that is used to set the “Terrain” game parameter. We specify this variable in another method:

public void SelectAndPlayFootstep()

{

switch (currentTerrain)

{

case CURRENT\_TERRAIN.GRAVEL:

PlayFootstep(1);

break;

case CURRENT\_TERRAIN.GRASS:

PlayFootstep(0);

break;

case CURRENT\_TERRAIN.WOOD\_FLOOR:

PlayFootstep(2);

break;

case CURRENT\_TERRAIN.WATER:

PlayFootstep(3);

break;

default:

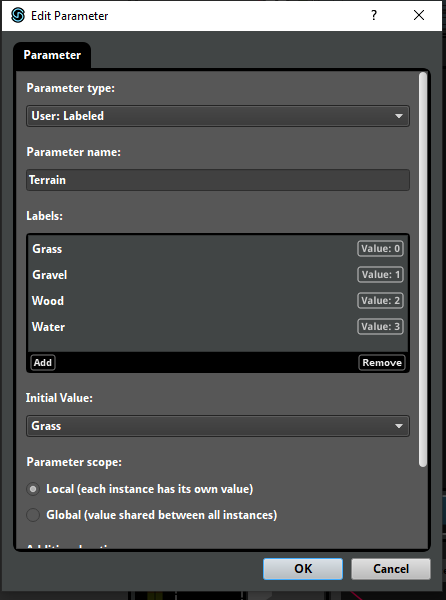
PlayFootstep(0);

break;

}

}

With this switch case we finally play a footstep sound. But how do we know that the int values correspond to certain ground types? In FMOD Studio, these values are displayed to the right of the label in the game parameter configuration:



FMOD Studio Labeled Parameter values

The complete script for the footsteps now looks like this:

public class PlayerFootsteps : MonoBehaviour {

private enum CURRENT\_TERRAIN { GRASS, GRAVEL, WOOD\_FLOOR, WATER };

[SerializeField]

private CURRENT\_TERRAIN currentTerrain;

private FMOD.Studio.EventInstance foosteps;

private void Update()

{

DetermineTerrain();

}

private void DetermineTerrain()

{

RaycastHit[] hit;

hit = Physics.RaycastAll(transform.position, Vector3.down, 10.0f);

foreach (RaycastHit rayhit in hit)

{

if (rayhit.transform.gameObject.layer == LayerMask.NameToLayer("Gravel"))

{

currentTerrain = CURRENT\_TERRAIN.GRAVEL;

break;

}

else if (rayhit.transform.gameObject.layer == LayerMask.NameToLayer("Wood"))

{

currentTerrain = CURRENT\_TERRAIN.WOOD\_FLOOR;

break;

}

else if (rayhit.transform.gameObject.layer == LayerMask.NameToLayer("Grass"))

{

currentTerrain = CURRENT\_TERRAIN.GRASS;

}

else if (rayhit.transform.gameObject.layer == LayerMask.NameToLayer("Water"))

{

currentTerrain = CURRENT\_TERRAIN.WATER;

}

}

}

public void SelectAndPlayFootstep()

{

switch (currentTerrain)

{

case CURRENT\_TERRAIN.GRAVEL:

PlayFootstep(1);

break;

case CURRENT\_TERRAIN.GRASS:

PlayFootstep(0);

break;

case CURRENT\_TERRAIN.WOOD\_FLOOR:

PlayFootstep(2);

break;

case CURRENT\_TERRAIN.WATER:

PlayFootstep(3);

break;

default:

PlayFootstep(0);

break;

}

}

private void PlayFootstep(int terrain)

{

foosteps = FMODUnity.RuntimeManager.CreateInstance("event:/Footsteps");

foosteps.setParameterByName("Terrain", terrain);

foosteps.set3DAttributes(FMODUnity.RuntimeUtils.To3DAttributes(gameObject));

foosteps.start();

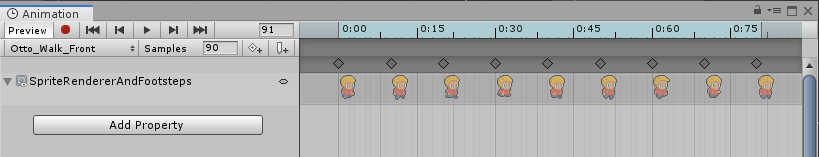
foosteps.release();

}

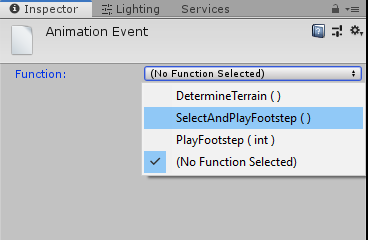
}

**Add footsteps sounds to animations using Animation Events**

Now open the Animation window, click the Player-GameObject (or a Child GameObject that contains the Animator component), and then select one of the walk or run animations from the Animation Clip drop-down list:

Unity Animation Clip

We right click below the timeline and above the keyframes on the free dark grey space. Select the **Create Animation Event** option. As a result, a small button/arrow (the Animation Event) should have been placed at the desired position/keyframe. In the inspector we now see that we can choose between different methods from a drop-down list:

Unity Animation Event select method

There we select the method SelectAndPlayFootstep() and repeat the process one more time. It is best to always select the keyframes where the player places the foot on the floor. In some situations, where this may not be quite clear, it is also sufficient if the two animation events are at the same distance from each other. It looks like that to me now:

Animation Events inside of an Animation Clip in Unity

Repeat this process for more animation clips. If you’ve done everything right, you should be able to hear your footsteps in Play Mode.