## Generating layer types

It’s not possible to generate the heightmap and splat/other layer types at the same time. So you first need to do the heightmap (version 2 there's automatic multi pass).

You need to clone the TerrainComposer\_save object in the scene. Then get the terraincomposer\_save script. Everything that's deactive will be deleted from the clone, to make generation faster. This is why you have to make a clone.

## Global Variables

#### // Can assign the TerrainComposer GameObject here

var TerrainComposer: GameObject;

#### // Can assign the terraincomposer\_save script from the TerrainComposer GameObject here

var tc\_script: terraincomposer\_save;

var TerrainComposerClone: GameObject;

var tc\_script2: terraincomposer\_save;

## Finding the TerrainComposer components by script

TerrainComposer = GameObject.Find("TerrainComposer");

tc\_script.TerrainComposer.GetComponent(terraincomposer\_save);

## Terrains

Your Terrains need to be in the TC Terrain List. The terrain index is place inside list, it starts with 0. So your first terrain is tc\_script.terrains[0].

#### //Active-Deactive

tc\_script.terrains[0].active = true/false;

## Layer outputs

Activate-Deactivate the layer outputs:

tc\_script.heightmap\_output = true/false;

tc\_script.color\_output = true/false;

tc\_script.splat\_output = true/false;

tc\_script.tree\_output = true/false;

tc\_script.grass\_output = true/false;

tc\_script.object\_output = true/false;

## Access components

### Layers

#### // Default layer level = 0, the layer number is displayed behind each layer.

tc\_script.prelayer[LayerLevel Index].layer[Layer Index]…….

We take layer level 0 and layer 0 as an example.

#### // Active-Deactive

tc\_script.prelayer[0].layer[0].active = true/false;

#### // Strength

tc\_script.prelayer[0].layer[0].strength = (float);

#### // Output

tc\_script.prelayer[0].layer[0].output =layer\_output\_enum.heightmap;

#### // Perlin Total

tc\_script.prelayer[0].layer[0].zoom = (float); // Zoom

tc\_script.prelayer[0].layer[0].offset = (Vector2);// Offset

## Filters and subfilters

To access filters and subfilters you need to know the index number, this you can see if you enable 'DataBase Restore' in the TC Menu -> Options -> Database Restore.

tc\_script.filter[filter index].

tc\_script.subfilter[subfilter index].

We take filter 0 is an example.

#### // Active-Deactivate

tc\_script.filter[0].active = true/false;

#### // Strength

tc\_script.filter[0].strength = (float);

#### // Input

tc\_script.filter[0].type = condition\_type\_enum.Height;

enums -> Height, Steepness, Direction, Image, Random, RandomRange, Always, Current, MaxCount, RawHeightmap.

#### // Output Filter

tc\_script.filter[0].output = condition\_output\_enum.add;

enums -> add, substract, change, multiply, devide, difference, average, max, min

#### // Output Subfilter

tc\_script.subfilter[0].output = subfilter\_output\_enum.max;

enums -> max, min, average, add, substract.

## Filter and Subfilter curves

tc\_script.filter[filter index].precurve\_list[curve index].

tc\_script.subfilter[subfilter index].precurve\_list[curve index].

The curve index number begins with 0.

We take filter 0 with curve 0 as an example

#### // Active-Deactive

tc\_script.filter[0].precurve\_list[0].active = true/false;

#### // Animation Curve

tc\_script.filter[0].precurve\_list[0].curve = (AnimationCurve);

#### // Curve Type

tc\_script.filter[0].precurve\_list[0].type = curve\_type\_enum.Normal;

enums -> Normal, Random, Perlin.

#### // Abs

tc\_script.filter[0].precurve\_list[0].abs = true/false;

#### // Perlin Curve

tc\_script.filter[0].precurve\_list[0].frequency = (float);

tc\_script.filter[0].precurve\_list[0]. offset = (Vector2);

#### // Invert the curve

tc\_script.filter[0].precurve\_list[0].set\_invert();

#### // Set default curve

tc\_script.filter[0].precurve\_list[0].set\_default();

## Make a Clone

Because TerrainComposer is multithread you need to make a clone, as all inactivate components will be removed to make generation faster.

TerrainComposerClone = Instantiate(TerrainComposer);

tc\_script2 = TerrainComposerClonse.GetComponent(terraincomposer\_save);

## Generate Preparing

#### // preparing the generation

tc\_script2.generate\_begin();

#### // set generate to active

tc\_script2.generate = true;

#### // the generate speed (int)

tc\_script2.generate\_step = 50;

#### // object placement speed (int), with break step it will return after placing 100 objects in a row.

tc\_script2.break\_x\_step = 100;

## Generate

function Update()  
{

if (tc\_script2)

{

If (tc\_script2.generate)

{

tc\_script2.generate\_output(tc\_script2.prelayers[0]);

}

else

{

#### // generation is ready so the clone can be destroyed

Destroy (TerrainComposerClone);

}

}

}