CloudsToy v1.0 for Unity 5.

How to test CloudsToy:

1.- Create a layer named **CloudsToy** in any layer number you want.
2.- Open the demo scene located in: Assets/Volumetric Clouds/Example Scenes/Test1

How to use CloudsToy in a new scene:

1.- Create a layer named **CloudsToy** in any layer number you want.
2.- Just drag the **CloudsToy Mngr** prefab into your scene.
3.- Hit Play and enjoy!

Comments:

The first variable "**Maximum Clouds**" will create internally all the clouds that CloudsToy is going to manage, they will be visible or invisible depending on the "Clouds Num" variable. Ideally, once you fine-tune your clouds to fit your needs, the Maximum Clouds and the Clouds Num variable should have the same value (so all the clouds you see are all the clouds the system is managing). ***IMPORTANT:*** *DO NOT change this variable in runtime, it will brake how CloudsToy works.*

The **Repaint** button is used because not all actions in the editor are going to change all the clouds in real-time mode (it could be too slow depending on the number of clouds that CloudsToy is managing), so if you want to be sure how actually the clouds looks like, hit the Repaint button just in case.

Soft Clouds is used to make sunrise type of clouds. They are very large, soft and almost transparent clouds.

Be aware that depending on the number of clouds, the type of render, the detail of the clouds (number of particles that the clouds have) and the size width, height and depth, the system's FPS may drop down dramatically. All this variables can be changed in the editor in real-time mode so be careful with what values you use.

All the clouds can have their own shadow (configurable by the user). To work properly, the clouds are stored in the 30th layer by default. You can create a new layer named **CloudsToy** in any layer number you want. CloudsToy will detect the new layer and use it .

The procedural texture can be used in the clouds in real-time if you choose Cloud Type == PT1 (Procedural Texture 1).

It can use two different noise generators to create the texture: simplenoise and perlin.

Using the same values when you run the application will not generate the exactly same textures twice. You can test it by clicking play button in Unity twice and you'll see the texture is different (sometimes completely different) using exactly the same texture parameters, that's the normal behavior not any estrange error.

Notice that Clouds Textures works drawing all the corners of the textures in black (see at the textures Unity provide in his Standard Assets to create smoke, for example), you can do the same using the Halo parameters (Halo and Radius) and seeing the result in the Procedural texture editor window and in the clouds themselves in the scene window (if you have chosen TypeCloud == PT1).

CloudsToy expect you to use a 64x64 or 128x128 texture size only, using greater values will cause a slow execution of the cloud system (depending on your computer) and different behaviors on Halo functions. Use mixed values like 64x128, will cause weird textures as result, so use that kind of texture sizes at your own risk!

Hit 'Save Texture' once you have fine tune your cloud texture. It will be saved at /Volumetric Clouds/Textures/Procedural. Do NOT erase that folder or CloudsToy will fail for sure. Once saved, use the new texture (Add and Blended) dragging them to the textures field in the clouds parameter. The blended textures are the same that Add ones but with an alpha channel (transparency) active. These are used depending on the type of render you want to use (Bright --> Add/Realistic --> Blended). Important: This feature to save textures will njot work in 'WebPlayer' targeted developments.

That's basically all the important stuff you have to know. Play with the cloud system a little bit and you'll see how fun is it. NOTE : With the appropriate skybox the clouds will look much better.

Jocyf 2015.
jocyf@jocyf.com

CloudsToy public variables

```
// Cloud main presets
public enum TypePreset { None = 0, Stormy = 1, Sunrise = 2, Fantasy = 3 }
public TypePreset CloudPreset = TypePreset.None;
public enum TypeRender { Bright = 0, Realistic = 1 }
public TypeRender CloudRender = TypeRender.Bright;
public enum TypeDetail { Low = 0, Normal = 1, High = 2 }
public TypeDetail CloudDetail = TypeDetail.Low;
public enum Type { Nimbus1=0, Nimbus2=1, Nimbus3=2, Nimbus4=3, Cirrus1=4, Cirrus2=5,
MixNimbus=6, MixCirrus=7, MixAll=8, PT1 = 9 }
public Type TypeClouds = Type.Nimbus1;

// Clouds shaders & projector material to be used.
public Shader realisticShader;
public Shader brightShader;
public Material projectorMaterial;

// Particles Advanced Settings
public float SizeFactorPart = 1;
public float EmissionMult = 1;

// Other cloud vars.
public bool SoftClouds = false;
public Vector3 SpreadDir = new Vector3(-1, 0, 0);
public float LengthSpread = 1;
public int NumberClouds = 100;
public Vector3 Side = new Vector3(1000, 500, 1000);
public Vector3 MaximunVelocity = new Vector3(-10, 0, 0);
public float VelocityMultipier = 1;
public float DisappearMultiplier = 1.5f;
public enum TypePaintDistr { Random = 0, Below = 1 }
public TypePaintDistr PaintType = TypePaintDistr.Below;
public Color CloudColor = new Color(1, 1, 1, 1);
public Color MainColor = new Color(1, 1, 1, 1);
public Color SecondColor = new Color(0.5f, 0.5f, 0.5f, 1);
public int TintStrength = 50;
public float offset = 0.5f;
public int MaxWithCloud = 100;
public int MaxTallCloud = 40;
public int MaxDepthCloud = 100;
public bool  FixedSize = true;
public enum TypeShadow { All = 0, Most = 1, Half = 2, Some = 3, None = 4 }
public TypeShadow NumberOfShadows = TypeShadow.Some;
public bool IsAnimate = true;
public float AnimationVelocity = 0;
```

The following public variables cannot be changed at runtime. CloudsToy create the clouds materials using this textures -and the selected shaders- when it initializes.

~~public Texture2D[] **CloudsTextAdd** = new Texture2D[6];~~
~~public Texture2D[] **CloudsTextBlended** = new Texture2D[6];~~