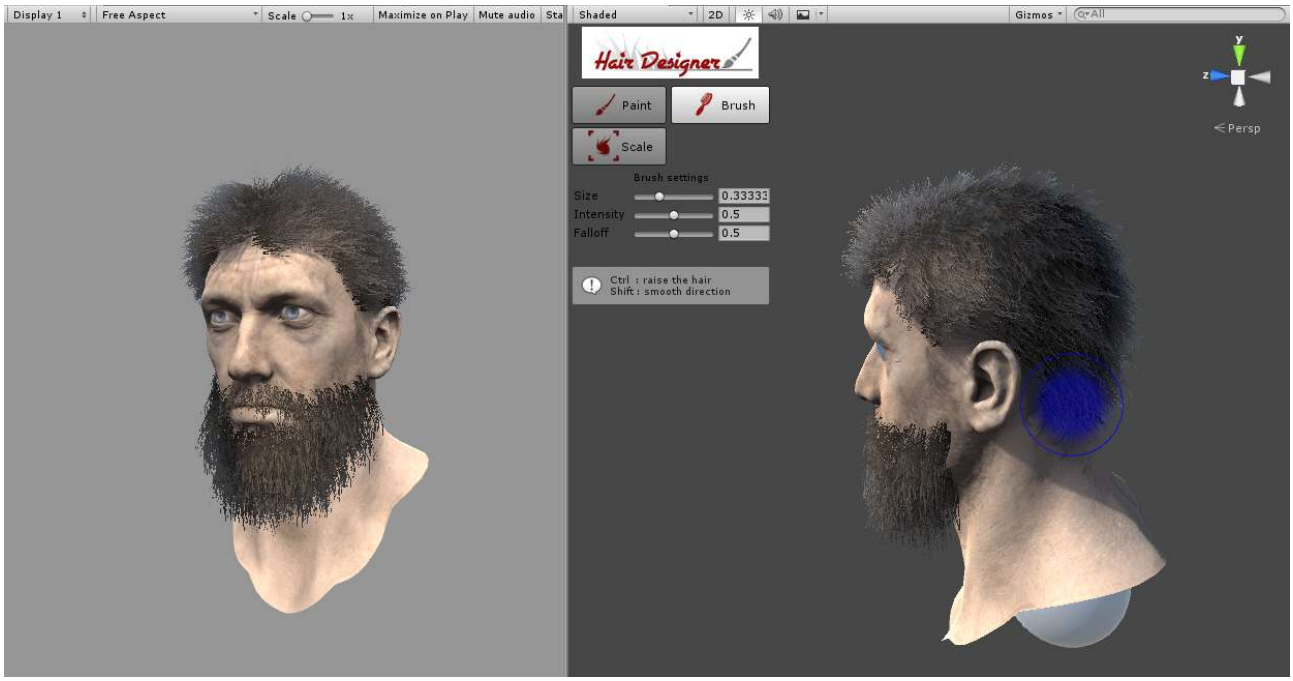
Documentation

Hair Designer

What is Hair Designer

Hair Designer is a new way for creating hair cut or fur within unity engine.
You can paint hair strands on the mesh and control the design.
Hair & fur are dynamic, motion and the gravity effects are computed by a compatible shader.



Timelaps video : https://www.youtube.com/watch?v=eLUjmbSEZwM
Forum : http://forum.unity3d.com/threads/hair-designer-dynamic-hair-fur-tool.431393/

How to start

First, select the gameobject with one of these components : MeshFilter or SkinMeshRenderer.
The mesh will be the base for the hair creation.

Add the HairDesigner component to the gameObject, from the menu :
Menu → Component → Hair Designer → Hair Designer

Layers

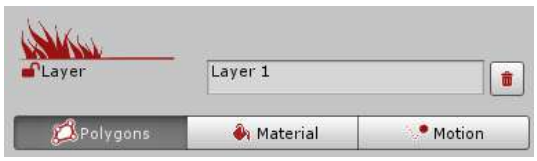
The ***Hair Designer*** extension is based on a layer system, each layer will generate a mesh including all the hair strands. A layer can be enable or disable with the checkbox.

Click on the layer for opening the edition mode.
In this mode, the name is editable and the delete button is available.

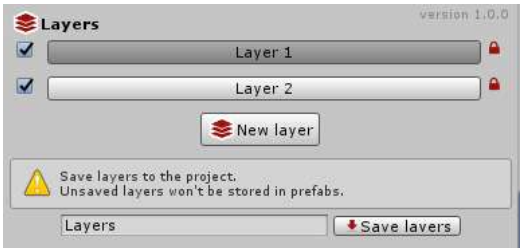
Each Layer is defined by a type :

- 'Short Hair/fur' : paint strand on the mesh surface.
- 'Long hair' : create hair using a curve tool.
- 'fur' : generate shells.

The edition mode is divided in 3 tabs :
Polygons, Material and Motion.



A layer must be locked when the design is over. Locking the layer allow you to save all the meshes to the project. If you need to create a prefab or to duplicate the gameObject, you must save the layers.



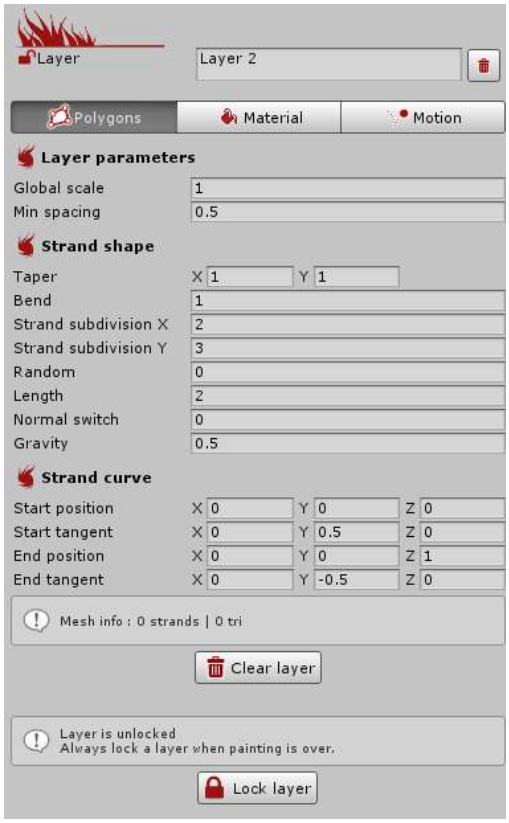
For locking/unlocking a layer, go to the Polygons tab.

If one of the layer has been modified, the 'save layer' panel will be displayed.

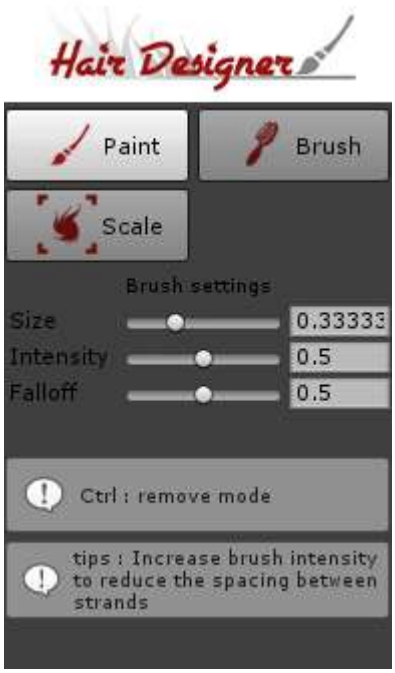
Layer : Short hair & fur

Polygons tab

The polygons tab contains the modeling options of the tool, and a panel is available in the sceneview with the painting tools.



- **Global scale** : the scale of the hair strands for the current layer.
- **Min spacing** : the minimum space between strands, this value is modulated by the Intensity of the paint brush.
- **Taper** : the taper value for the bottom(x) and the top (y) of the strands.
- **Strand subdivision** : each strand is defined by a quad mesh. Increasing subdivision set a better look and feel, but add more triangles to the overall hair cut.
- **Random** : a random amplitude factor for each strand.
- **Length** : the length of the hair strands
- **Normal switch** : switch normal from the normal of the face (0) to the tangent of the curve (1). Use this parameter to change how the light react on the hair.
- **Gravity** : the gravity factor
- **Strand curve** : the curve fo the hair strands.
- **Clear layer** : remove all strand from the layer.
- **Lock Layer** : lock the polygon creation and compute the final mesh.



Paint tool :

This is the tool for creating hair strand on the target surface.
Strand spacing is defined by 'Min spacing' parameter and the intensity of the brush.

Ctrl : Remove the hair strands .

Brush tool :

This tool set the hair strand direction.

Ctrl : Raise the hair strands.
Shift : Smooth the hair strands' orientation.

Scale tool :

Scale up the hair strands.

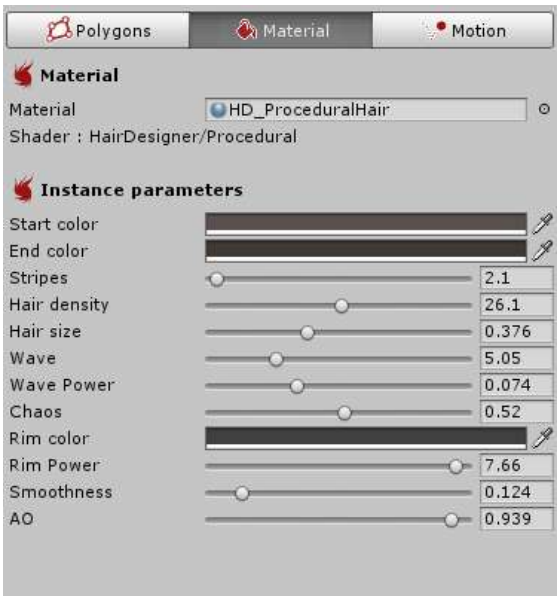
Ctrl : Scale down.
Shift : Smooth the hair strands.

Material tab

The material tab define the material used for the mesh and all the instance parameters defined in the shader.
The default material is 'HD_ProceduralHair' with the 'HairDesigner/ProceduralHair' shader.
Any material can be used, but only a compatible shaders will have instance parameters.

Drag & drop a material in the 'Material' field. If the shader is compatible, the 'Instance parameters' panel will be available.

Tips : Switch between Polygon tab and Material tab for tuning the hair cut design, then lock the layer.



Motion tab

Hair Designer is a dynamic hair system, the motion system requires a motion zone. The motion zone encapsulates the hair mesh and is attached to a gameObject. For SkinMeshRenderer, motion zones have to be attached to a bone. The motion zone detect the motion of its parent and apply it to all the hair strands in its range. You can add many motion zones (up to 50) for a more complex effect.

Position & hierarchy :

Drag & drop the parent transform or choose it in the bone list (SkinMesh renderer only)

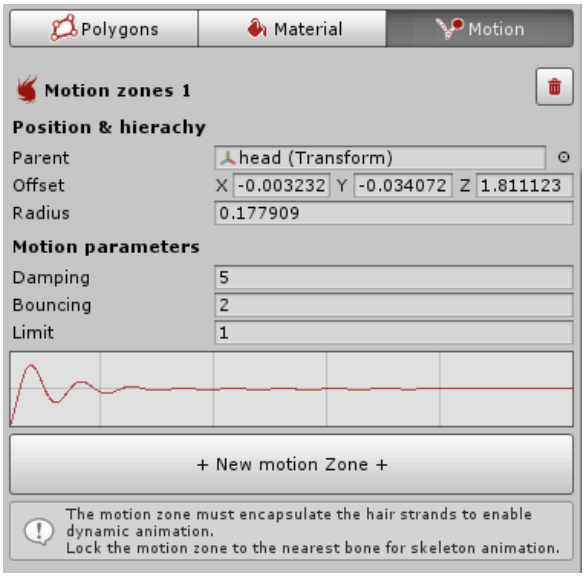
Offset and Radius are defined by moving the zone in the scene view.

Motion parameters :

The motion is defined by a curve, defined by the damping and the bouncing parameters.

The limit parameter fix the motion amplitude.

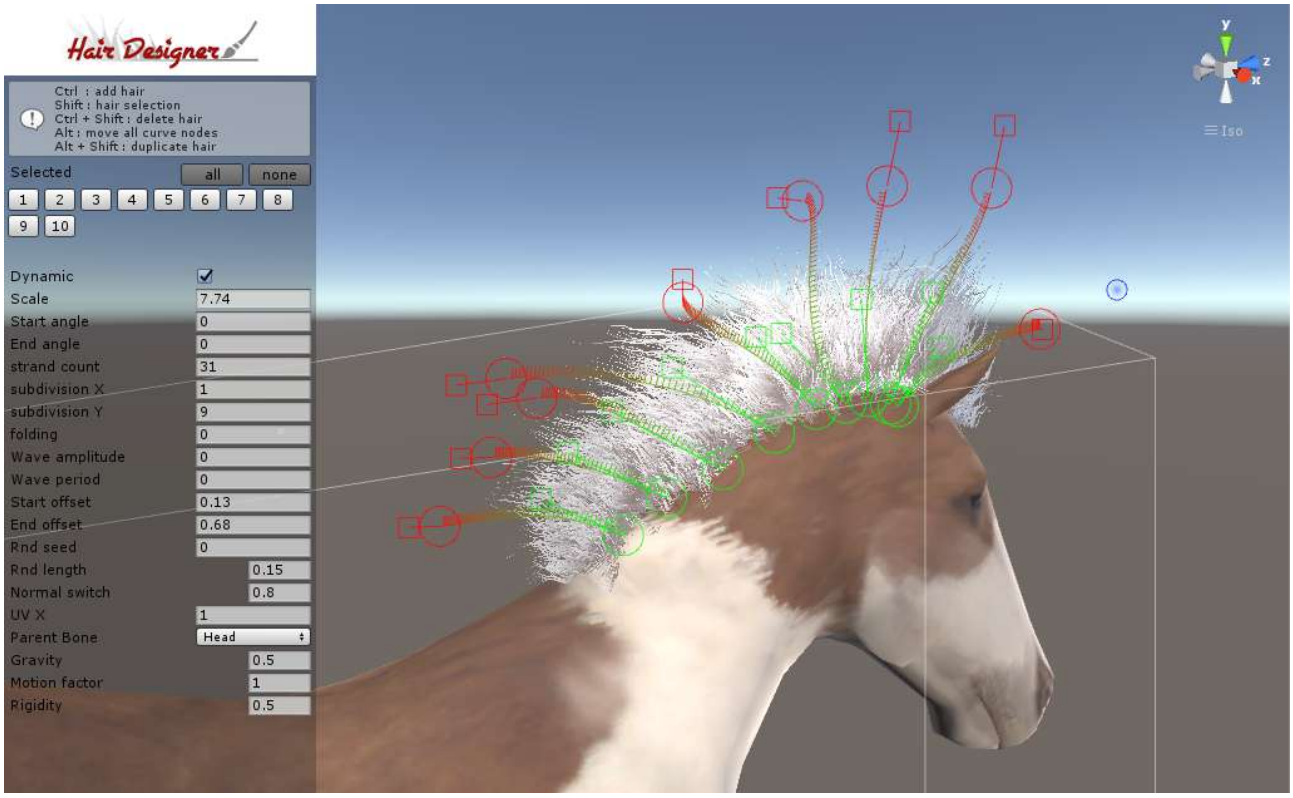
Note : a section on the curve represent 1 second.



Layer : Long hair

Polygons tab

This layer is composed of polygons defined by curves with procedural parameters.
Each curve can be modified in the sceneview.



The tools functions are :

- **Ctrl** : Add hair curve to the model.
- **Shift** : show hair selection buttons in the scene view .
- **Ctrl + Shift** : show delete buttons for each curve.
- **Alt** : move all node of the current curve.
- **Alt+ Shift** : show the duplicate buttons.

Note : the selection button of an unselected curve is hidden when a curve is on the back, change the camera view to see them.

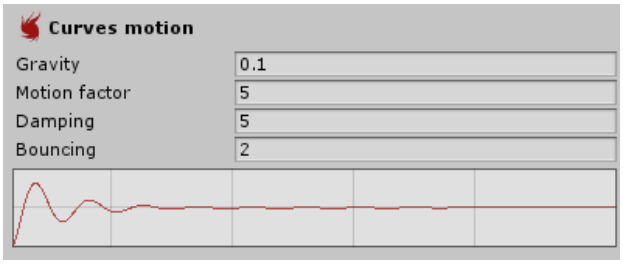
When several curves are selected, you can change their global settings in the scene view, or change each parameter individually in the inspector tab.

Curve parameters :

- **Dynamic** : if checked, the curve will move acording to its parent's motion.
- **Scale** : the scale of the polygons on the curve.
- **Start angle**: Start angle of the strands.
- **End angle** : End angle of the strands.
- **stand count** : number of strands following the curve.
- **subdivision X** : subdivision for the width.
- **subdivision Y** : subdivisions along the curve.
- **Folding** : use this parameter for wavy hair, acording to the 'wave' and 'wave power' parameters in the material tab.
- **Wave amplitude** : amplitude of the wave along curve.
- **Wave period** : period of the wave along curve.
- **Start offset** : Start offset in a circle at the start of the curve.
- **End offset** : End offset in a circle at the end ot the curve.
- **Rnd seed** : Random seed of the hair strand, use this to change the look of the strands.
- **Normal switch** : switch normal from the normal of the face (0) to the tangent of the curve (1). Use this parameter to change how the light react on the hair.
- **UV X** : duplicate the UV,x for simulating more strands on the same strand.
- **Parent Bone** : lock the curve on a specific bone. Try to lock to the closest one.
- **Gravity** : how gravity will be applied on the curve, from 0 to 1.
  note : this value is multiplied by the Gravity parameter in the motion tab.
- **Motion factor** : how the strand will react to its parent's motion.
  note: this value is multiplied by the motion factor in the motion tab.
- **Rigidity** : how the base of the hair (start tangent) react to motion.
  0 → full motion / 1 → base static

Motion tab

The motion tab is similar to the short hair motion tab, there's some additionnals settings for curves motion.



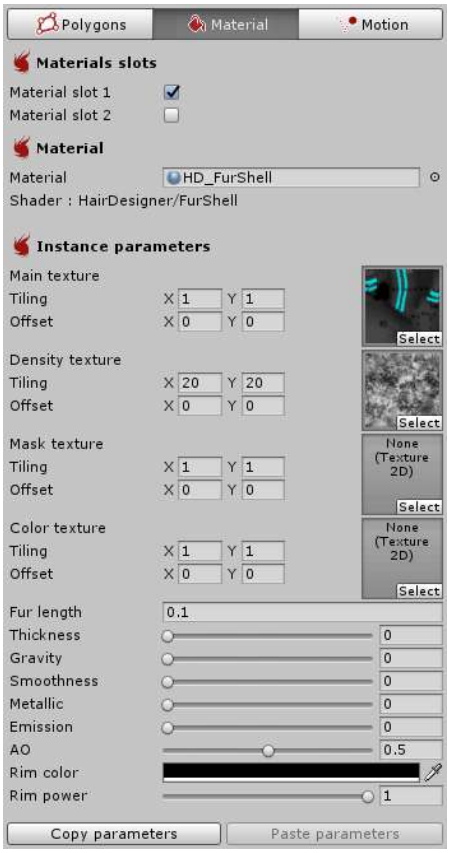
- **Gravity** : how gravity (Physics.gravity) will be applied on the curve. If the gravity is greater than the parent's movement, it will override the small variations of motion.
  note : this value is multiplied by the Gravity parameter for each curve.
- **Motion factor** : how the strand will react to its parent's motion.
  A value of 1 means that the hair will follow the exact movement of it's parent, most of the time we want to amplify the fx.
  note: this value is multiplied by the motion factor for each curve.

Layer : fur

The fur layer is a powerfull and easy tool for generating an amazing fur effect to your characters.

Material tab

The material slot enable or disable the fur effect for each material of the renderer.



Main texture : the fur texture, it could be the original mesh texture, or another one.

Density texture : this texture generate the fur implantation, the tiling defined the thickness of each strand.

Mask texture : this texture contains the length and the direction of each strand on the mesh. The texture is generated by the painting tools.

Color texture : this texture contains additionnal colors. The texture is generated by the painting tools.

Fur length : he maximum fur length.

Thickness : extra thickness

Gravity : gravity factor

Smoothness : PBR smoothness factor

Metallic : PBR Metallic factor

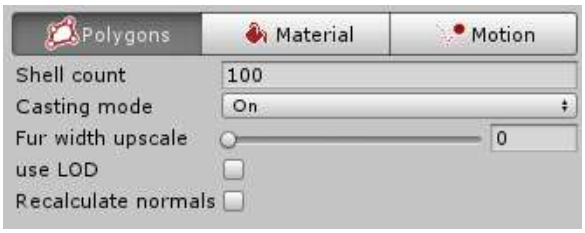
Emission : Emission factor

AO : ambiant occlusion for the root of the strands

Rim color & Rm power : settings for Rim effect.

Polygons tab

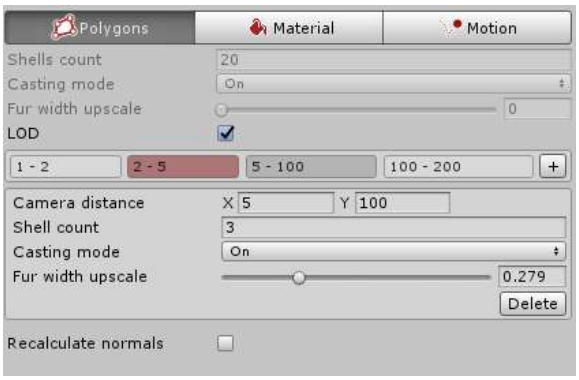
This layer use the shell technic, so you can change the shell count to increase fur quality.
A high shell count will use more ressources, try to set it to the lower as possible.



The casting mode defined if the type of shadows for the fur, it could impact the FPS depending of your lighting settings.

LOD system

When the camera is far from the gameobject, the shell count can be decrease to improve performances. The fur width can be upscaled for keeping the look of the entire fur with a lower shell count. When the LOD is enabled, the parameters are defined according to the distance of the camera. The current LOD (red) depend of the sceneview camera, you can adjust each LOD group by zooming the camera in the sceneview.



The '+' button add a new LOD group, each LOD parameters can be edited and are linked to the last sceneview rendered.

Note : If the original mesh cast shadows, you don't need to enable shadows for the fur when the camera is too far.

The 'recalculate normal' parameter can be checked if there's some problem with the original mesh normals.

Painting tools : Fur mask

This layer is driven by textures that will be used by the shader for the fur generation.

First you have to create a mask that will be stored to the project. This mask define the length of the fur and the directions of the strands.

When the mask is created, the paint tool is enable. You can paint the fur on the model itself.
The brush can be clamped between 2 values for a better control of the design.
The toggle 'lock no mask' avoid the brush to paint on empty zones.

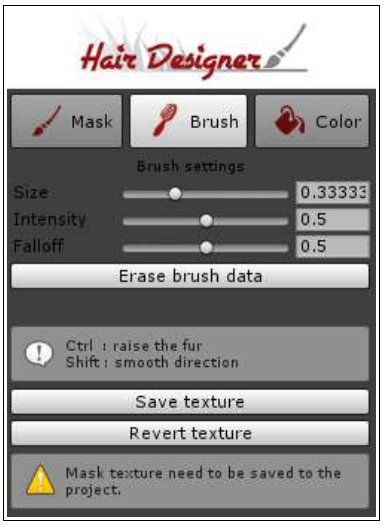
It's possible to fill all the mask with the min or max values, or use a texture mask, based on the model UV.

Brush options :

Ctrl : decrease the fur length.
Shift : smooth the fur length in the brush area.

The texture need to be save to the project.

Painting tools : Fur brush



The brush tool move fur direction.

Brush options :

Ctrl : raise the fur.
Shift : smooth the fur direction in the brush area.

Note : The brush tool works with the model geometry, so it would be more precise on a high poly mesh.

Painting tools : Fur color

Fur color is fully customizable, the 'Replace/Mix' parameter define how the color is combine with the main texture, you can mix with it or override it.

Motion tab

The motion tab is similar to the short hair motion tab

Custom Shader

The compatible shaders uses the Hair Designer vertex program function.

Here a way to include it in your custom shader :
1. Change the name to 'HairDesigner/shaderName'
2. Add the include file : 'HairDesigner.cginc'
3. Add the vertex program function :

```
void vert (inout appdata_full v, out Input o)
{
	HairDesigner(v);
}
```

or include the function in your existing vertex function.

How to include instance parameter to the Hair Designer UI :

1. Create a class with the name : HairDesignerShader + ShaderName

```
public class HairDesignerShaderFire : HairDesignerShader
{
    //HERE THE MEMBERS
    public float _NbFlame = 5f;
    public Color _Color1 = Color.yellow;

    public override void UpdatePropertyBlock(ref MaterialPropertyBlock pb)
    {
      //HERE THE PROPERTY BLOCK UPDATE
       pb.SetColor("_Color1", _Color1);
       pb.SetFloat("_NbFlame", _NbFlame);
    }
}
```

2. Create the associated Editor class : HairDesignerShader + ShaderName + Editor

```
[CustomEditor(typeof(HairDesignerShaderFire))]
public class HairDesignerShaderFireEditor : Editor
{
    public override void OnInspectorGUI()
    {
        HairDesignerShaderFire s = target as HairDesignerShaderFire;

        GUILayout.Label("Render", EditorStyles.boldLabel);
        s._NbFlame = EditorGUILayout.Slider("Details", s._NbFlame, 1, 50);
        s._Color1 = EditorGUILayout.ColorField("_Color1", s._Color1);
    }
}
```