V1.0 - Local Documentation

Online Documentation

Support: support@marian-brinkmann.com

MatEdit is a tool which you can use to develop eye-catching custom editors for shaders very fast and without any effort.

Furthermore, it adds intuitive Color Gradients and Animation Curves to your material inspector. This means you can create and edit such textures directly in the material inspector! This will speed up your iterations and prototypes recognizably.

MatEdit supports grouping in the editor in several ways and allows you to reset groups separately. Moreover, you can add custom context menu options via code.

If you plan to distribute your shader including the GUI a distribution dll is available.

Index

Index **2**

How to Start **3**

Possibilities 3

Distribute with MatEdit **4**

How to Distribute 4

Special Fields **4**

Further Information **5**

How to Start

In order to support you taking your first steps with MatEdit we provide this "How to Start" guide.

Simply follow the given steps in order to create your custom material inspectors:

1. Right-click the shader you want to enhance with a custom material inspector and select "Create > MatEdit > Create ShaderGUI"
2. Open the editor script generated in the folder "Editor" relative to the shader file.
3. Select the presented method under the SetScope() method call.
4. Use any MatGUI method to create new fields for the material inspector.

If you plan to add it **manually** or in an **already existing ShaderGUI** follow the upcoming instructions instead:

1. Create a new C# script and name it as you wish. It is recommended to call it relatable to your shader.
2. Open the script in the script editor of your choice.
3. Remove all MonoBehaviour related functions etc.
4. Use “MB.MatEdit” and “UnityEditor”.
5. Inherit from “ShaderGUI” and implement the method “OnGUI”
6. At this point it is recommended to use “MatGUI.SetScope” to define the target of all upcoming operations.
7. Use any MatGUI-Field method you want.
8. To bind this GUI to the shader open your shader and insert before the last braces: CustomEditor “YourGUIScript”

Possibilities

Giving you a more user friendly and readable approach for GUI code is not the only task MatEdit tries to fulfill.

By using the special fields you can enable the user to change Color Gradients and Animation Curves directly in the material inspector. These fields generate a texture which can be read in the shader.

In order to guarantee the smooth functionality of MatEdit in future updates it is recommended to use the cg-inclusions provided with MatEdit. These cginc files can be placed manually in your shader's folder or can be used project wide by using the installation button in the about window. The cginc files provide a method for converting the textures back into Color Gradients or Animation Curves.

Distribute with MatEdit

Creating a readable and organized material inspector is fine but if you want to publish your shader including the GUI on the Asset Store or other distribution platforms you have to deliver MatEdit together with your own scripts.

MatEdit allows you to use the MatEdit.dll for the distribution but it is not allowed to distribute the source code without permission.

How to Distribute

In order to make the distribution as easy and convenient as possible you can use the context menu as follows:

Right-click the shader you want to distribute. Select: Create > MatEdit > Create Distribution.

In the upcoming window choose the folder in your project you want to store the distributable version of your shader in.

Special Fields

MatEdit often utilizes data as textures to the shader which uses another user interface. To read the information in the shader simply include the MatEditCG.cginc file.

For this process there are two ways:

1. Go the the About window via: Window > MatEdit > About - and install the cg files by clicking the install button.
2. Copy the MatEditCG.cginc, GradientCG.cginc or CurveCG.cginc file depending on what you need.

In the shader you can sample an Animation Curve as follows:

```
float process = 0.5;
float result = sampleCurve(_AnimationCurve, process);
```

Very important! The AnimationCurveField can only contain data ranging from (x: 0 - 1, y: 0 - 3). Currently the rgb channels are used for: r = 0 - 1, g = 1 - 2, b = 2 - 3.

It is possible that the conversion from Animation Curve to Texture is changes. By using the given CG-includes you are safe in future updates.

You can sample a Color Gradient like this:

```
float process = 0.5;
float4 result = sampleGradient(_ColorGradient, process);
```

Further Information

In order to provide you with the most recent news about MatEdit further information can be found in our online documentation. There you will find script references and examples, as well.