Materializer

Version 1.0

Introduction:

Welcome and thank you for your purchase of the Materializer PBR Texturing Engine for Unity. While PBR/PBS is a relatively new concept to may Unity Users, we feel that tools to help game designers in the PBR workflow are necessary to help push the envelope of modern game development. By using Materializer, you are on the path to adding visually stunning and physically accurate texturing to your next project.

What is Materializer?

Materializer is a Plugin for Unity that will assist you in texturing your game elements like Characters, Environments, Particle Systems and anything that uses a Mesh Renderer in your Scene. Materializer provides Physically Balanced Texture sets for various shader frameworks. This means that Materializer will provide you with the textures you need to feed into your shader like Diffuse/Albedo, Normal, Specular, Gloss/Roughness, Ambient Occlusion, Transparency and Metallic.

All of the textures generated by Materializer are Physically balanced according to your chosen shader framework, meaning that the textures provided will enforce PBR constraints like Conservation of Energy and Metallic/Dielectric balance. Materializer will output up to 4K (4096x4096) textures.

Materializer ships with over 100 base PBR material definitions for you to use in your projects. There is also an additional package registered users can download that contains a collection or pre-compiled materials that can be used if your scene.

What Materializer isn't:

One of the most commonly mistaken assumptions about Materializer is that it is a Shader or Shader Framework for Unity. While Materializer works closely with many PBR/PBS shading frameworks, it in and of itself is not a Shader. Materializer is responsible for providing inputs (Textures) for your existing PBR Shader.

System Requirements:

In order use Materializer you will need to make sure that your development environment meets the following requirements.

- Unity 4.5 or Higher (Unity Pro is not required to run the tool itself, but all PBR shaders require Linear Color Space in order to work, which is only available in Unity Pro)
- A PBR Shader
- 4 Gigs of RAM (8 recommended)
- Windows 7 or higher
- OSX Lion or Higher

Supported Shader Frameworks:

- Alloy Physical Shader Framework by Rust Ltd.
- Lux Physically based shader framework by Forst
- Jove Physically based Shading System by Jove Software
- Standard Shader for Unity 5 by Unity Technologies
- Skyshop IBL Shaders by Marmoset.

A few things to know about PBR/PBS and IBL:

If you are new to Physically Based Rendering and Image Based Lighting there are a few things you need to be made aware of so you completely understand what is happening. Firstly the goal of PBR is consistency in the rendering fidelity of your project no matter what the lighting condition is. Simply put this means that materials like Gold should look like Gold in outdoor lighting as well as indoor or no lighting. This is achieved by accurately describing the Physical material of Gold to your shader with PBR balanced color maps.

Based on the laws of the Conservation of Energy, a photon of light hits the surface of an object and then a certain percentage of that light is absorbed into the surface thus causing the Albedo (Diffuse) color of your surface. The other portion of the light is then reflected back into the world thus causing your Specularity. Since energy can neither be created nor destroyed, there must always be a perfect balance between the percentages of light absorbed and reflected. So if 25% of the light hitting your surface is absorbed into the diffuse/albedo portion of your material, then the additional 75% must then be reflected.

In addition to the law of conservation of energy, you must also consider how the light being reflected back in the Specular channel should look. Which directions should that reflected light scatter? This is why we use a Roughness/Gloss map. Roughness/Gloss tells your shader what the Microsurface of your material looks and feels like.

Choosing a PBR Shader:

Each PBR Shader Framework is a little different the next. For example:

- Alloy uses a reversed map style for Gloss/Roughness.
- Lux and Jove both have an integrated Image Based Lighting (IBL) solution but Alloy does not.
- Skyshop will not work with Jove or Lux.
- Lux and Jove both have a free version of their Shader framework
- Alloy has a direct integration with Skyshop
- Some shaders require Deferred rendering
- Alloy, Lux and Jove require Unity Pro to take advantage of Linear Color Space
- Unity 5 Standard Shader may not require Pro (unknown at this time)

Installation:

In order to use Materializer you will need to install the .unitypackage that you hopefully purchased from the AssetStore. Simply open the installer package and click the import button to begin the installation process. It may take a few minutes to import the full package, as the bulk of the content are graphic files used as material brushes. Once the installation is complete, you are ready to begin using Materializer.

Basic Workflow:

The basic workflow of Materializer is to first define the regions of your geometry, which you wish to apply certain textures to. There are 2 main way of doing this the first being to have your 3D artist create your models in such a way that they are actually modular and you can apply solid texture to various components as you see fit. An example of this would be making a riffle that consists of many sub meshes like the stock, the action, the trigger, the barrel and sights. In this way you can create solid material textures and apply them to the various parts of the riffle; like wood for the stock, steel for the barrel and aluminum for the action.

While the first method will give you a bit more flexibility to play around with various materials and even make adjustments at runtime, your game will suffer some in terms of performance due to the higher amount of draw calls required to render each material separately in each frame. A higher performance alternative exists and is in fact the desired workflow for Materializer.

The 2nd workflow consists of creating color masks on the UV plane(s) of your model and importing that into Materializer so it can process each region individually. Then once you are happy with the material design and effects, merge those maps into a single set of outputs. This workflow has been around for quite a while and is popular in many other texturing applications. The one thing that typically holds up production on this workflow is a lack of access to the original 3D models or UV wireframes, which you can use as a guide to hand, paint in applications like Photoshop.

In order to assists with overcoming this hurdle we have provided a UV wireframe tool which can in most cases reverse engineer the UV map of your 3D models and let you export them out to .png format.

In either case of the workflows that you chose, the next step is to assign the textures you have created to Materials/Shaders in unity. Because we support so many shader frameworks, it is a bit beyond the scope of Materializer to map these out for you automatically. However we do have guides to working with each shader framework on our YouTube channel located at:
https://www.youtube.com/watch?v=V6IKiix3X60

Steps for creating a new texture set:

Now that you are ready to begin creating your first texture set, you will need to follow the steps below to create your new texture project.

1. Open Unity and make sure that you have installed the Materializer package.
2. Click on the Window/Materializer/New Project menu item.
3. In the New Project window provide a Name for your new Project.
4. Specify the Mesh that you want to bind your material to.
5. Provide your Color Mask file. (You can use a solid color squared image for solid materials)*
6. Provide your base normal map file.*
7. Provide your Base Ambient Occlusion map (optional, but preferred).*
8. Specify your project resolution.**
9. Select your target shader framework.
10. Adjust your Color Matching settings (Used for color masks with antialiasing).
11. Adjust your Saturation settings (used to ignore noise).
12. Adjust your Edge width settings (used for edge based effects).
13. Click the Bake button to begin the process of creating your project.
14. Once finished, you will be notified that the new project is complete.

*For information on how to create Color/Diffuse Masks, Normal Maps and Ambient Occlusion Maps for your mesh check the documentation from your 3D modeling software provider. Normal Maps should be Tangent Space Normal Maps. Color Masks should be solid non-antialiased colors. In most cases your 3D designer will be able to provide these files for you.

**As a general rule, the higher the resolutions of your textures, the longer it will take so bake them. While Materializer does support 4K textures, you should be aware that compiling 4K textures could take a relatively long time.

Now that your new project has been created the next steps are going to cover how to materialize your projects.

1. Open the main Materializer window by clicking the Window/Materializer/Materializer menu item.
2. Choose the project you would like to work on from the “Select Project” list.
3. Choose the Map you would like to modify from the Maps list
4. Choose the Effect(s) you would like to add to that map
5. If you setup more than one effect, chain them together in the sequence you wish.
6. Click the Paint button.
7. Repeat steps 3-6 for each map you want to use in your project.
8. Once you have finished defining all of your materials, click the Merge button.
9. Once the final merge is completed, you will want to start assigning your textures to your material/shader.*

   *You can find your compiled textures in Unity by browsing to Assets/Materializer/Textures/Temp/Your Project Name/ folder.

Additional Tips:

1. You can view the files that have been created for an individual map by right clicking on the map in the Maps list and selecting the file you want to view.
2. You can filter your plugin list by typing a search phrase in the Filter Plugins text box
3. You can save you effect chain for later use by clicking the Save button.
4. You can recall the compile history of individual maps by accessing the Properties window by right clicking on the map in question and selecting Properties. In this window you will see a list of previously compiled versions of your Map, simply click on the version you want to reload and it will reload those settings into the design surface of the main Materializer window.

Additional Materials:

The base Materializer package ships with over 100 pre configured materials. While this will suffice to many texturing situations, we will offer many additional material packs for purchase and free download via our Material Store.