http://www.informit.com/articles/article.aspx?p=2162089&seqNum=2

-------------------------------------------------------------------

http://answers.unity3d.com/questions/20461/making-a-texture.html

here above the best is .png

next best is psd files

textures are just image files use

use Paint Shop Pro to make textures, since textures are just image files. Like I said before, if Paint Shop Pro exports/saves to PNG format,

-------------------------------------------------------------------------------------------------------------------------------------------------------

What is a PBR shader?

Physically Based Rendering (PBR) is a method of shading and rendering that provides a more accurate representation of how light interacts with surfaces. It can be referred to as Physically Based Rendering (PBR) or Physically Based Shading (PBS).Mar 17, 2017



The Beginner’s Guide to Physically Based Rendering in Unity

<http://blog.teamtreehouse.com/beginners-guide-physically-based-rendering-unity>

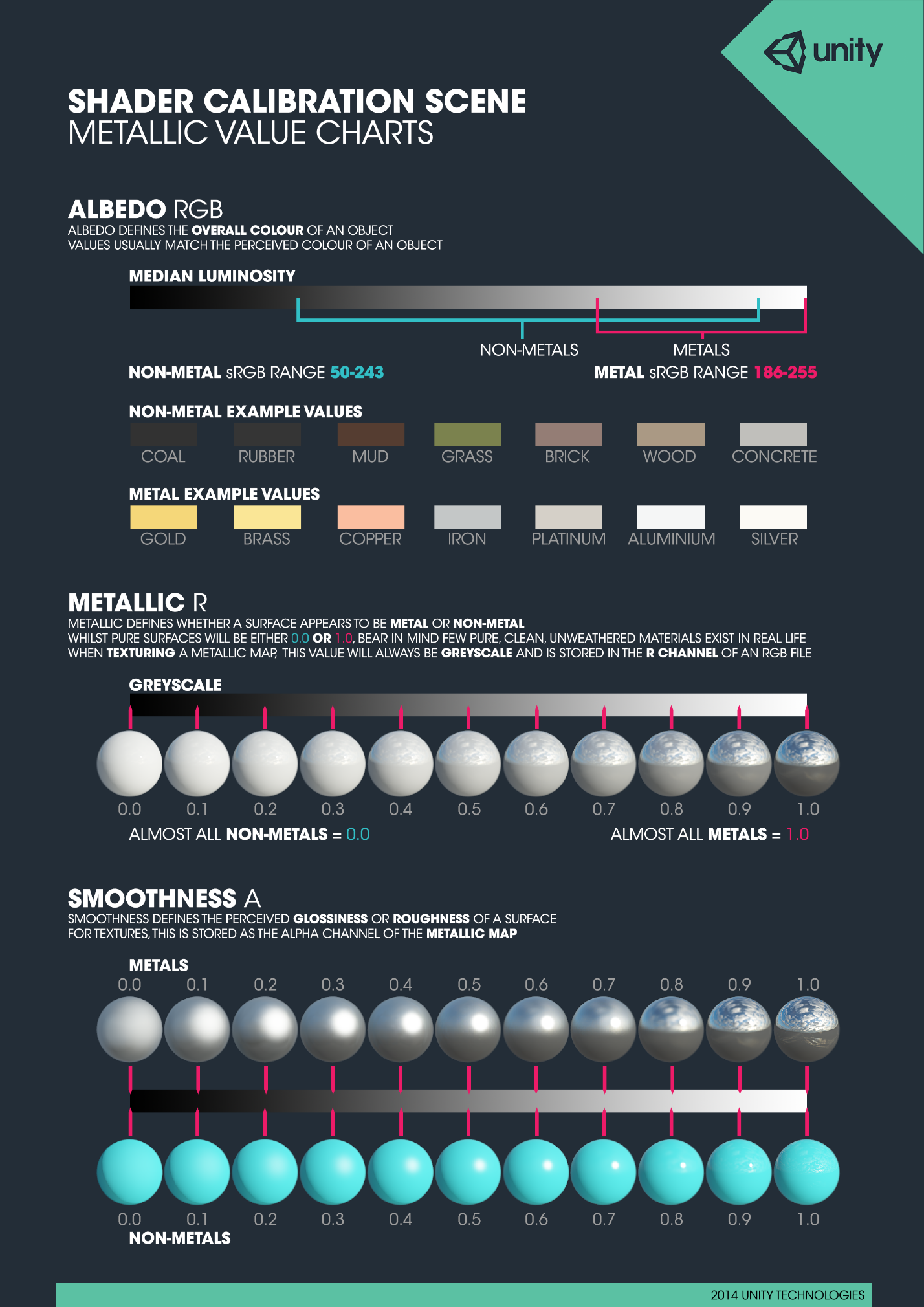
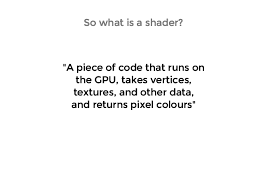
1. The principles of PBR were pioneered by Disney and Pixar for use in offline rendering

2 but advances in hardware and software have made PBR possible in real-time rendering for games.

3 Thus far, various forms of PBR have been implemented in both the Unity and Unreal game engines, as well as in major art tools like Autodesk Maya and Allegorithmic Substance Designer.

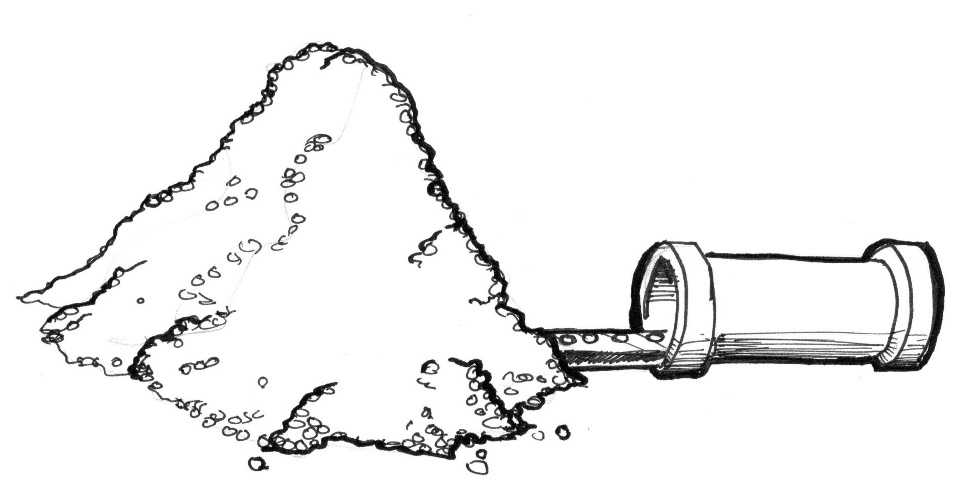
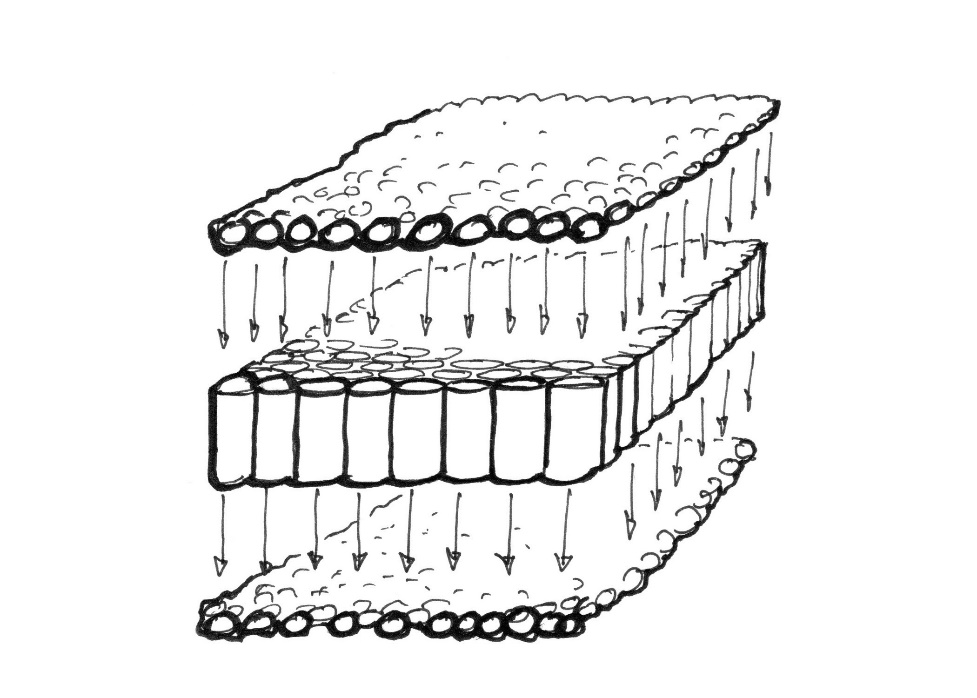
1. With PBR, assets will look the way you’d expect when there are changes in lighting or in the environment.
2. Almost all PBR renderers use some form of image-based lighting, which allows assets to pick up colors from the surrounding environment
3. his is because PBR uses scientifically calibrated color for known real-world materials, which reduces guess-work and trial-and-error.
4. M*aterial* is the combination of a shader and texture maps. In the case of Unity’s default Standard shader (and most other PBR shaders), there are several texture map slots (also sometimes called “channels”). The three most important ones are:

* Albedo
* Metallic
* Normal

1. when creating a material like aluminium, gold, or coal, you can simply refer to a PBR chart and get the scientifically accurate value. For example, here is Unity’s official calibration chart.
2. 
3. What is a shader?
4. 
5. two most important shaders are the vertex shader and the fragment shader.
6. The vertex shader is called once per vertex. So if you have a triangle to be rendered then the vertex shader is called three times, one for each corner.  For simplicity we can imagine that a fragment is a pixel on the screen, and therefore the fragment shader is called for every resulting pixel.
7. The two shaders have different roles. The vertex shader is primarily used to transform the 3D model data into a position in the 3D world as well as map the textures or the light sources, again using transformations. The fragment shader is used to set the color of the pixel, for example by applying the color to the pixel from a texture map.
8. If you noticed each vertex is handled independently from the other vertexes. The same is also true for the fragments.
9. What this means is that the GPU could run the shaders in parallel, and in fact, that is what it does.
10. The vast majority of mobile GPUs have more than one shader core.
11. <http://www.androidauthority.com/what-is-a-gpu-gary-explains-693542/>
12. ------------------------------------------------------------------------------------------------------

{{{{{{{{{

->Video games and other graphic applications require a lot more processing power than other programs.

1. Because of their graphic content they have to do huge numbers of pixel-by-pixel operations
2. Every single pixel on the screen needs to be computed, and in 3D games geometries and perspectives need to be calculated as well.
3. Let's go back to our metaphor of the pipes and tasks. Each pixel on the screen represents a simple small task. Individually each pixel task isn't an issue for the CPU, but (and here is the problem) the tiny task has to be done to each pixel on the screen! That means in an old 800x600 screen, 480,000 pixels have to processed per frame which means 14,400,000 calculations per second! Yes! That’s a problem big enough to overload a microprocessor. In a modern 2880x1800 retina display running at 60 frames per second that calculation adds up to 311,040,000 calculations per second. How do graphics engineers solve this problem?
4. 
5. This is when parallel processing becomes a good solution. Instead of having a couple of big and powerful microprocessors, or *pipes*, it is smarter to have lots of tiny microprocessors running in parallel at the same time. That’s what a Graphic Processor Unit (GPU) is.
6. 

*GPU*

1. Picture the tiny microprocessors as a table of pipes, and the data of each pixel as a ping pong ball. 14,400,000 ping pong balls a second can obstruct almost any pipe. But a table of 800x600 tiny pipes receiving 30 waves of 480,000 pixels a second can be handled smoothly. This works the same at higher resolutions - the more parallel hardware you have, the bigger the stream it can manage.
2. Another “super power” of the GPU is special math functions accelerated via hardware, so complicated math operations are resolved directly by the microchips instead of by software. That means extra fast trigonometrical and matrix operations - as fast as electricity can go.

}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}

# {{{{{{{{{{{

# Optimizing graphics performance

<https://docs.unity3d.com/Manual/OptimizingGraphicsPerformance.html>

}}}}}}}}}}}}}}}}}}

{{{{{{{{{

Cookbook for shaders

<file:///C:/Users/ptech-user/Downloads/Unity%20Shaders%20and%20Effects%20Cookbook.pdf>

}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}

{{{{{{{{{{{{

# Virtual Realism with Unity 5

[http://archvirtual.com/2015/06/05/virtual-realism-with-unity-5/#](http://archvirtual.com/2015/06/05/virtual-realism-with-unity-5/)

}}}}}}}}}}}}}}}}}}}}}}}}}

{{{{{{{{{{{{{{{

# Commonly Used Software and Techniques for Visual Effects in Films and Games

<http://www.admecindia.co.in/blog/commonly-used-software-and-techniques-visual-effects-films-and-games>

A [**matte painting**](http://www.admecindia.co.in/graphic-design-portfolio/matte-paintings-in-photoshop) as its name describes is painted representation of something like set, location, or landscape. It is very useful to create illusion of an environment that is impossible in real life and very expensive too for film makers and game develop

### [Rotoscoping](http://www.admecindia.co.in/sites/default/files/rotoscoping-in-movies.JPG?width=80%25&height=80%25)

[**Rotoscoping**](http://www.admecindia.co.in/post-production-portfolio/rotoscoping) is the technique of physically modifying film or video footage frame by frame at a time. The frames can be painted on subjectively to make exclusively animated impacts like lightning or light-sabres, or followed to make realistic traditional style animation or to deliver hold-out mattes for compositing components in a scene and, all more recently, to create depth maps for stereo transformation.

}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}

{{{{{{{{

http://www.creativebloq.com/3d/10-greatest-vfx-movies-all-time-21619264

# The 10 greatest VFX movies of all time

### 06. The Matrix

Few films have had as much influence on movie effects as The Matrix

Released in spring 1999, The Matrix was the surprise science fiction hit that beat The Phantom Menace to the finish line in the race for the Academy Award for Visual Effects. It launched the career of first-time visual effects supervisor John Gaeta, then just 34, and goes down in history as the film that raised the bar for the choreography of fight sequences and reinvented cinematography.

Its most iconic scene is a frozen moment that has become known as 'bullet time', in which Neo (Keanu Reeves) dodges bullets fired at him by an agent, while the camera circles around. The sequence still captivates today.

Bullet time blew our minds in '99, and it still looks amazing today

Gaeta's team trained a circular array of 122 still cameras on Keanu Reeves, then triggered them in sequence. Because cameras located on one side of Reeves were visible to those on the other, Gaeta needed a way of generating photorealistic sets so the camera could be removed from frame.

Gaeta and Manex VFX supervisor Kim Libreri found the answer at Siggraph 1997 in 'The Campanile Movie', a short film by Paul Debevec, George Borshukov and Yizhou Yu. Photographs of buildings were reprojected onto their CG models using new, best-fit algorithms. The result was the birth of virtual cinematography.

The idea of thinking in code has never been better represented

"All my friends who worked on the film were enablers who allowed us to take risks," remembers Gaeta. "That was very important to me at that time, and gave me confidence in everything else I've done since. The Wachowskis give their designers quite a bit of creative freedom, and they really engaged, encouraged and inspired."

Reflecting on the film in 2007, Gaeta noted that there were many shots he'd still like to tweak. "There are a lot of shots that, in the light of what you can do today, are pretty crude. But they still represent ideas the filmmakers were trying to represent." To judge from The Matrix's placement in the poll, they're ideas that endure to this day.

### 05. Interstellar

Christopher Nolan doesn't do things by halves in this Oscar-winning outing

Collaborating once more with director Christopher Nolan, Interstellar challenged Double Negative to visualise the un-visualisable: realistic alien worlds, a mathematically accurate black hole, and the Tesseract, a four-dimensional space with time as a physical dimension. Theoretical physicist Kip Thorne provided the maths, the studio's artists delivered the visuals, and Hollywood awarded them an Oscar.

Lots of science went into this. But more importantly, it looks really cool

For many viewers, the most memorable shot from the movie is the depiction of the black hole Gargantua, for which Double Negative needed to show the realistic behaviour of the black hole and a wormhole, right down to the lighting – or lack thereof.

For this, Double Negative was lucky to have, in Oliver James, a chief scientist with a first-class degree in physics from the University of Oxford. To process the equations Kip Thorne – who also acted as executive producer on the movie – had written to describe light paths around a black hole, James and his team wrote a new physically based renderer: DnGR (Double Negative General Relativity). It enabled artists to generate realistic images of the hole and its gravitational lens by setting three key parameters: rate of spin, mass and diameter.

Theoretical physicist Kip Thorne provided the maths for this VFX masterclass

The work broke new ground, both artistically and scientifically: a paper on the research written by James, Thorne, CG supervisor Eugenie von Tunzelmann and VFX supervisor Paul Franklin was recently published in the American Journal of Physics – which promptly called for the movie to be shown in school science lessons to help explain general relativity.

In an interview with BBC News, Christopher Nolan commented that reactions like these were the "ultimate goal" of the movie. "We hoped that by dramatising science and making it... entertaining for kids we might inspire some of the astronauts of tomorrow," he said.

# }}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}

# {{{{{{{{{{{{{{{{{

# What’s Your VR + Architecture Strategy?

# <http://archvirtual.com/2016/11/29/whats-vr-architecture-strategy/>

# }}}}}}}}}}}}}}}}}}}}}}

# {{{{{{{{{{{{{{{{{{{{{ Theory of achieving Photo-realism in Unity 5 !

# <http://www.vivekc.com/theory-of-achieving-photo-realism-in-unity-5/>

# }}}}}}}}}}}}}}}}}}}}}}}}}}

# {{{{{{{{{{

Photo-realistic" quality rendering between Unity and Unreal 4

<https://forum.unity3d.com/threads/photo-realistic-quality-rendering-between-unity-and-unreal-4.262181/>

}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}

{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{

# Using detail textures for extra realism close-up

<https://unity3d.com/learn/tutorials/topics/graphics/using-detail-textures-extra-realism-close>

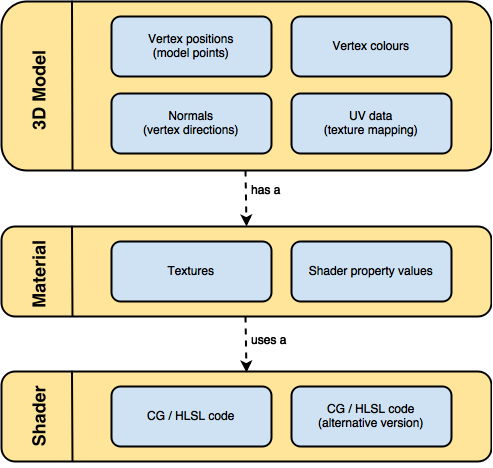
}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}}

{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{

# A Gentle Introduction to Shaders

[[[[[[[[[[[[[[[[[[[[[[https://unity3d.com/learn/tutorials/topics/graphics/gentle-introduction-shaders]]]]]]]]]]]]]]]]]]]]]

[Alan Zucconi's site](http://www.alanzucconi.com/2015/06/17/surface-shaders-in-unity3d/).



1. Unity supports two different types of shaders: suface shaders and fragment and vertex shaders.
2. Shader "MyShader"
3. {
4. Properties
5. {
6. // The properties of your shaders
7. // - textures
8. // - colours
9. // - parameters
10. // ...
11. }
12. SubShader
13. {
14. // The code of your shaders
15. // - surface shader
16. // OR
17. // - vertex and fragment shader
18. // OR
19. // - fixed function shader
20. }
21. }
22. You can have multiple SubShader sections, one after the other.
23. They contain the actual instructions for the GPU. Unity will try to execute them in order, until it finds one that is compatible with your graphics card.