

# glTF-Tutorials

[Previous: Meshes](#)[Table of Contents](#)[Next: Simple Material](#)

## Materials

### Introduction

The purpose of glTF is to define a transmission format for 3D assets. As shown in the previous sections, this includes information about the scene structure and the geometric objects that appear in the scene. But a glTF asset can also contain information about the *appearance* of the objects; that is, how these objects should be rendered on the screen.

There are different possible representations for the properties of a material, and the *shading model* describes how these properties are processed. Simple shading models, like the [Phong](#) or [Blinn-Phong](#), are directly supported by common graphics APIs like OpenGL or WebGL. These shading models are built on a set of basic material properties. For example, the material properties involve information about the color of diffusely reflected light (often in the form of a texture), the color of specularly reflected light, and a shininess parameter. Many file formats contain exactly these parameters. For example, [Wavefront OBJ](#) files are combined with `.MTL` files that contain this texture and color information. Renderers can read this information and render the objects accordingly. But in order to describe more realistic materials, more sophisticated shading and material models are required.

### Physically-Based Rendering (PBR)

To allow renderers to display objects with a realistic appearance under different lighting conditions, the shading model has to take the *physical* properties of the object surface into account. There are different representations of these physical material properties. One that is frequently used is the *metallic-roughness-model*. Here, the information about the object surface is encoded with three main parameters:

- The *base color*, which is the “main” color of the object surface.
- The *metallic* value. This is a parameter that describes how much the reflective behavior of the material resembles that of a metal.
- The *roughness* value, indicating how rough the surface is, affecting the light scattering.

The metallic-roughness model is the representation that is used in glTF. Other material representations, like the *specular-glossiness-model*, are supported via extensions.

The effects of different metallic- and roughness values are illustrated in this image:

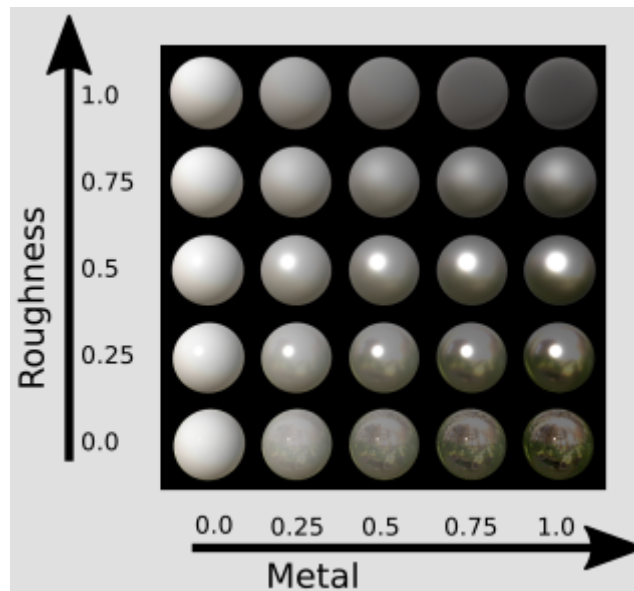


Image 10a: Spheres with different metallic- and roughness values.

The base color, metallic, and roughness properties may be given as single values and are then applied to the whole object. In order to assign different material properties to different parts of the object surface, these properties may also be given in the form of textures. This makes it possible to model a wide range of real-world materials with a realistic appearance.

Depending on the shading model, additional effects can be applied to the object surface. These are usually given as a combination of a texture and a scaling factor:

- An *emissive* texture describes the parts of the object surface that emit light with a certain color.
- The *occlusion* texture can be used to simulate the effect of objects self-shadowing each other.
- The *normal map* is a texture applied to modulate the surface normal in a way that makes it possible to simulate finer geometric details without the cost of a higher mesh resolution.

glTF supports all of these additional properties, and defines sensible default values for the cases that these properties are omitted.

The following sections will show how these material properties are encoded in a glTF asset, including various examples of materials:

- [A Simple Material](#)
- [Textures, Images, and Samplers](#) that serve as a basis for defining material properties
- [A Simple Texture](#) showing an example of how to use a texture for a material
- [An Advanced Material](#) combining multiple textures to achieve a sophisticated surface appearance for the objects

This site is open source. [Improve this page.](#)