**standard_surface**

**Class**
Shader

**Output**
RGB

**Synopsis**

The *Standard Surface* shader is a physically-based shader capable of producing many types of materials. It includes a diffuse layer, a specular layer with complex Fresnel for metals, specular transmission for glass, subsurface scattering for skin, thin scattering for water and ice, a secondary specular coat, and light emission.

This shader models a material consisting of ten components that are layered and mixed hierarchically as illustrated in the diagram below. The properties of individual components can vary across the surface.

![Schematic illustration of the idealized material model that standard_surface emulates. Horizontal stacking of components represents statistical mixture and vertical stacking represents layering. Components marked with an asterisk are optional.](image)

**Material Types**
By default, the parameters are appropriate for materials such as plastic, wood or stone. By setting a few key parameters to 1, different types of materials can be quickly created:

- **Metalness**: gold, silver, iron, car paint.
- **Transmission**: glass, water, honey, soap bubble.
- **Subsurface**: skin, marble, wax, paper, leaves.
- **Thin Walled**: paper, leaves, soap bubble.

Parameter values between 0 and 1 may be used to create more complex materials that are a mix of basic material types.

### Energy Conservation

Standard Surface is energy conserving by default. All its layers are balanced so that the amount of light leaving the surface does not exceed the amount of incoming light. For example, as a surface is made more metallic and the specular layer contribution is increased, the diffuse layer contribution is reduced accordingly to ensure energy conservation.

When using layer weights or colors with values higher than 1, energy conservation is broken. It is discouraged to create such materials, as they will not behave predictably under different lighting, and may lead to increased noise and poor rendering performance.

#### base

The base color weight (default is 0.8).

#### base_color

The base color sets how bright the surface is when lit directly with a white light source (intensity at 100%). It defines which percentage for each component of the RGB spectrum which does not get absorbed when light scatters beneath the surface. Metal normally has a black or very dark base color, however, rusty metal/s need some base color. A base color map is usually required.
**diffuse_roughness**

The base component follows an Oren-Nayar reflection model with surface roughness. A value of 0.0 is comparable to a Lambert reflection. Higher values will result in a rougher surface look more suitable for materials like concrete, plaster or sand.

0 (default) 0.5 1

**specular**

The specular weight. Influences the brightness of the specular highlight.
**specular_color**
The color the specular reflection will be modulated with. Use this color to 'tint' the specular highlight. You should only use colored specular for certain metals, whereas non-metallic surfaces usually have a monochromatic specular color. Non-metallic surfaces normally do not have a colored specular.

**specular_roughness**
Controls the glossiness of the specular reflections. The lower the value, the sharper the reflection. In the limit, a value of 0 will give you a perfectly sharp mirror reflection, while 1.0 will create reflections that are close to a diffuse reflection. You should connect a map here to get variation in the specular highlight.
The ‘microscopic’ features of a surface affect the diffusion and reflection of light. This ‘micro surface’ detail has the most noticeable effect on specular reflections. In the diagram below, you can view parallel lines of incoming light commence to diverge when reflected from rougher surfaces when each ray hits a part of the surface with a different orientation. In summary, the rougher the surface becomes, the more the reflected light will diverge or appear ‘blurred.’

The brightness of the Specular highlight is automatically linked to its size due to the Standard Surface Surface shader’s energy conserving nature. In the example below, all of the materials are reflecting the same amount of light, but the rougher surface is spreading it out in multiple directions. However, with low amounts of roughness, the surface is reflecting a more concentrated amount of light.

To get variation in the highlights of the surface, a map should be connected to the Specular Roughness. This will influence not only the brightness of the highlight but also its size and the sharpness of the environmental reflection.

You may need to connect the file texture to a Range shader to see any results in the Specular Roughness.
The specular roughness affects both specular reflection and refraction. There is also a Transmission Extra Roughness parameter to add some additional roughness for refraction if required. You can, however, use Coat to create a rough reflection layer over a sharp refraction.

specular_IOR

The IOR parameter (Index of Refraction) defines the material’s Fresnel reflectivity and is by default the angular function used. Effectively the IOR will define the balance between reflections on surfaces facing the viewer and on surface edges. You can see the reflection intensity remains unchanged, but the reflection intensity on the front side changes a lot.
Specular IOR with Transmission

The default value of 1.0 is the refractive index of a vacuum, i.e., an object with IOR of 1.0 in empty space will not refract any rays. In simple terms, 1.0 means 'no refraction'. The Standard Surface shader assumes that any geometry has outward facing normals, that objects are embedded in the air (IOR 1.0) and that there are no overlapping surfaces.

Using a very high IOR value can look quite similar to Metalness. It looks the same if you set the Base Color to the Specular Color and the Specular Color to black. The difference is that you get an extra reflection at the edges, with the Specular Color controlling the edge tint. The metal fresnel works the same as in the new complex IOR shader, with the artistic parameters.

You should normally use IOR for materials like plastic, glass, or skin (dielectric fresnel) and Metalness for metals (conductive fresnel with Complex IOR). The other reason is that Metalness is easier to texture since it's in the 0..1 range, and using textures from applications like Substance painter works best when using Metalness rather than IOR.

Normals

When rendering refractive surfaces, it is very important that the normals of the geometry face in the right direction. In the example below (left), you can see the difference between normals that are facing in the right direction (outward), versus those that are facing inwards (incorrect). This is especially important when rendering surfaces with double-sided thickness, such as glass. However, with air bubbles in glass (below center), the reverse is true. The bubble geometry normals should be reversed and the bubbles should be combined with the glass geometry. Normal direction is equally important when rendering single sided surfaces such as a car windscreen (right).
specular_anisotropy

Anisotropy reflects and transmits light with a directional bias and causes materials to appear rougher or glossier in certain directions. The default value for Anisotropy is 0, which means ‘isotropic.’ As you move the control towards 1.0, the surface is made more anisotropic in the U axis.

0 (default) 0.6 0.9

Anisotropy is suitable for materials that have a clear brush direction such as brushed metal which has tiny grooves in which form a ‘stretched’ anisotropic reflection.

Many small discs form together to create an effect which is the anisotropic highlight.

If you see any black where there should be refraction, you may not have a high enough Transmission Ray Depth value (found in the Ray Depth section in the render settings). The default value is eight, which is sufficient for most cases.
Anisotropic reflections are suitable for brushed metal effects such as in the example below:

You may notice faceting appear in highlights when using Anisotropy. It is possible to remove the faceted appearance by enabling smooth subdivision tangents (via Arnold subdiv_smooth_derivs parameter). Take into account this requires a subdivision iteration of at least one in the polymesh to work.
Faceting seen in the anisotropic highlight


Increase Subdivision Iterations to remove anisotropic faceting.

**specular_rotation**

The rotation value changes the orientation of the anisotropic reflectance in UV space. At 0.0, there is no rotation, while at 1.0 the effect is rotated by 180 degrees. For a surface with brushed metal, this controls the angle at which the material was brushed. For metallic surfaces, the anisotropic highlight should stretch out in a direction perpendicular to the brushing direction.

It is possible to assign textures to the rotation. When doing so, it is advisable to avoid texture filtering. This means disabling MIP-mapping and disabling the magnification filter, which by default is set to "smart bicubic." One way is to set the mipmap_bias of the image node to a strong negative value, like -8, which means "use 8 MIP levels higher resolution than usual."

**metalness**

With metalness 1.0 the surface behaves like a metal, using fully specular reflection and complex fresnel.
The metal appearance is controlled using the **Base Color** (facing) and **Specular Color** (edge tint) parameters. These are automatically translated to physical and values, to achieve the same look but with easily tweakable and texturable colors. Some examples of real-world values for metals can be found below.

<table>
<thead>
<tr>
<th>Metal</th>
<th>Base Color</th>
<th>Specular Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aluminium (Al)</td>
<td>0.912 0.914 0.920</td>
<td>0.970 0.979 0.988</td>
</tr>
<tr>
<td>Copper (Cu)</td>
<td>0.926 0.721 0.504</td>
<td>0.996 0.957 0.823</td>
</tr>
<tr>
<td>Gold (Au)</td>
<td>0.944 0.776 0.373</td>
<td>0.998 0.981 0.751</td>
</tr>
</tbody>
</table>

For a perfectly sharp and mirror-like reflection, increase **Metalness** to 1 and reduce **Specular Roughness** to 0. The **Base Weight** should also be set to 1.

When **Metalness** is enabled, the **Specular Weight** and **Specular Color** only control the edge tint, while the **Base** is still affected by roughness.
<table>
<thead>
<tr>
<th>Element</th>
<th>Value 1</th>
<th>Value 2</th>
<th>Value 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iron (Fe)</td>
<td>0.531</td>
<td>0.512</td>
<td>0.496</td>
</tr>
<tr>
<td></td>
<td>0.571</td>
<td>0.540</td>
<td>0.586</td>
</tr>
<tr>
<td>Lead (Pb)</td>
<td>0.632</td>
<td>0.626</td>
<td>0.641</td>
</tr>
<tr>
<td></td>
<td>0.803</td>
<td>0.808</td>
<td>0.862</td>
</tr>
<tr>
<td>Mercury (Hg)</td>
<td>0.781</td>
<td>0.779</td>
<td>0.779</td>
</tr>
<tr>
<td></td>
<td>0.879</td>
<td>0.910</td>
<td>0.941</td>
</tr>
<tr>
<td>Nickel (Ni)</td>
<td>0.649</td>
<td>0.610</td>
<td>0.541</td>
</tr>
<tr>
<td></td>
<td>0.797</td>
<td>0.801</td>
<td>0.789</td>
</tr>
<tr>
<td>Platinum (Pt)</td>
<td>0.679</td>
<td>0.642</td>
<td>0.588</td>
</tr>
<tr>
<td></td>
<td>0.785</td>
<td>0.789</td>
<td>0.784</td>
</tr>
<tr>
<td>Silver (Ag)</td>
<td>0.962</td>
<td>0.949</td>
<td>0.922</td>
</tr>
<tr>
<td></td>
<td>0.999</td>
<td>0.998</td>
<td>0.998</td>
</tr>
</tbody>
</table>
Metalness values between 0.0 and 1.0 can be used to texture surfaces like rusted iron, where different areas of the surface can have more reflective clean metal and more diffuse rust. PBR metalness maps from applications like Substance Painter can be connected to this parameter.
transmission

Transmission allows light to scatter through the surface, for materials such as glass or water.
transmission_color

This filters the refraction according to the distance traveled by the refracted ray. The longer light travels inside a mesh, the more it is affected by the Transmission Color. Therefore green glass gets a deeper green as rays travel through thicker parts. The effect is exponential and computed with Beer's Law. It is recommended to use light, subtle color values.

If you use a fully saturated color like (1, 0, 0), the way that is interpreted is that all red light is allowed to pass through, and no green and blue light. Transmission Color values near 0 make the interior of the mesh very dense to block all light, and then setting the Depth multiplier to a small value like 0.001 might not make much of a difference because the Depth is large anyway.
If this value has a color and shadows tinted with that color are required, then disable Opaque for the mesh that has been assigned the Standard Surface shader. In the example below, you can see that with Opaque enabled the rays cannot pass through the sphere. Whereas with Opaque disabled, the rays can pass through the sphere and absorb the color set by the Transmission Color, thereby creating the effect of colored shadows.

Note that Transmission Color will not work for single sided geometry unless Thin Walled is enabled.
transmission_depth

Controls the depth into the volume at which the transmission color is realized. Increasing this value makes the volume thinner, which means less absorption and scattering. It is a scale factor so that you can set a transmission color and then tweak the Depth to be appropriate for the size of your object.

The effect of increasing the Transmission Depth can be seen in the animation below. Note that a Transmission Scatter color has also been used in this case.
Transmission Scatter is suitable for any liquid that is fairly thick or where there is enough of it for scattering to be visible, such as a deep body of water or honey. If you have a glass of water, there is not that much scattering, however, for an ocean, it is required. Other examples include materials like ice, opalescent glass or milky glass.

*Depth* is scene scale dependent and can have a dramatic effect on its appearance. The *Transmission Color* and *Depth* control transmittance/absorption, and that depends on the object scale. So for a small object to see anything you might need to set a quite low *Depth*, or for a big object a high one. If you cannot see the effect of *Depth*, then you may need to check the size of your scene.

When the scene scale is too small, the *Transmission Color* (orange) appears incorrect (left image) with a *Transmission Depth* of 1. Lowering the *Transmission Depth* fixes it (right image). It is recommended to model to real-world scale to avoid these situations.
transmission_scatter_anisotropy

The directional bias, or anisotropy, of the scattering. The default value of zero gives isotropic scattering so that light is scattered evenly in all directions. Positive values bias the scattering effect forwards, in the direction of the light, while negative values bias the scattering backward, toward the light.

transmission_dispersion

Specifies the Abbe number of the material, which describes how much the index of refraction varies across wavelengths. For glass and diamonds, this is typically in the range of 10 to 70, with lower numbers giving more dispersion. The default value is 0, which turns off dispersion. The chromatic noise can be reduced by either increasing the global Camera (AA) samples or the Refraction samples.
Transmission Dispersion is ideal for gemstone materials like diamond.

**transmission_extra_roughness**

Adds some additional blurriness of refraction computed with an isotropic microfacet BTDF. The range goes from -1 to 1, where 0 means no roughness. It is computed as

\[ \text{Transmission Roughness} = \text{Specular Roughness} + \text{Transmission Extra Roughness}. \]

Negative values result in a lower roughness for transmission than for reflection.
transmit_aovs

When enabled, Transmission will pass through AOVs. If the background is transparent, then the transmissive surface will become transparent so that it can be composited over another background. Light path expression AOVs will be passed through so that for example a diffuse surface seen through a transmissive surface will end up in the diffuse AOV. Other AOVs can also be passed straight through (without any opacity blending), which can be used for creating masks for example.

Alpha masks (rollover for beauty)
Sub-Surface Scattering (SSS) simulates the effect of light entering an object and scattering beneath its surface. Not all light reflects from a surface. Some of it will penetrate below the surface of an illuminated object. There it will be absorbed by the material and scattered internally. Some of this scattered light will make its way back out of the surface and become visible to the camera. This is known as 'sub-surface scattering' or 'SSS'. SSS is necessary for the realistic rendering of materials such as marble, skin, leaves, wax, and milk. The SSS component in this shader is calculated using a brute-force raytracing method.

You must ensure that the geometry normals are pointing in the correct direction, otherwise, SSS will not render correctly.

SSS is important when replicating realistic materials such as plastics, for example:
**subsurface**

The 'blend' between diffuse and subsurface scattering. When set to 1.0, there is only SSS, and when set to 0 it is only Lambert. In most cases, you want this to be 1.0 (full SSS).

![SSS disabled](image1) ![SSS enabled](image2)

0 (default) 0.5 1

**subsurface_color**

The color used to determine the subsurface scattering effect. For example, replicating a skin material would mean setting this to a fleshy color.
Skin Color

Generally, you would connect a color texture map to the Subsurface Color, and set the subsurface radius to a value appropriate for skin (mostly a higher value for the red channel assuming red mostly comes from blood). The radius can also be textured if desired. Veins would be part of the color texture map, and that not only affects the color but also how the subsurface scattering behaves. Standard Surface has a mix between Base and SSS. It is your choice if you want to use diffuse with skin. It can be used to e.g. mix in a different texture with a mask for e.g. face paint, or to vary sharpness by connecting the same texture to both the Base Color and Subsurface Color and changing the blend factor.

subsurface_radius

The approximate distance up to which light can scatter below the surface, also known as “mean free path” (MFP). This parameter affects the average distance that light might propagate below the surface before scattering back out. This effect on the distance can be specified for each color component separately. Higher values will smooth the appearance of the subsurface scattering, while lower values will result in a more opaque look.

The lighter the color, the more light is scattered. A value of 0 will produce no scattering effect:
Increasing the radius value can radically change the appearance of the material, from looking like leather to marble. SSS is very scale dependent. You will need to adjust the radius multiplier depending on the size of your model. If you were to render using the default SSS settings, you might get something that does not look correct. Alternatively, adjusting the scene scale can have similar results.

Instead of distributing all of the colors with the same amount, you can also choose different radius values for each of the RGB colors. For example, a material like clay or skin should have a higher red radius than green and blue.

When rendering skin, you should use a value like 1.0, 0.35, 0.2, indicating that red should scatter deepest and green and blue less. This would replace the three-layer workflow (deep, mid and shallow layers) for skin where you would make the deep layer red to indicate that deep should scatter with a larger radius. Setting scatter_radius.R to a larger value would correspond to that.
The images below show the effect when increasing the red color of the radius. Notice the colored 'fringing' effect around the edges of the circles. The same effect occurs when Gaussian blurring the red channel of the source image in a compositing package.

**subsurface_scale**

Controls the distance that the light is likely to travel under the surface before reflecting back out. It scales the scattering radius and multiplies the SSS Radius Color.
If the scene is in meters, then *SSS Scale* can be set to 0.01 to specify the *SSS Radius* in centimeters. For example for skin, the *SSS Radius* could be 0.37cm, 0.14cm, 0.07cm.

**subsurface_anisotropy**

Henyey-Greenstein Anisotropy coefficient between -1 (full back-scatter) and 1 (full forward-scatter). The default is 0 for an isotropic medium, which scatters the light evenly in all directions, giving a uniform effect. Positive values bias the scattering effect forwards, in the direction of the light, while negative values bias the scattering backward, toward the light.

This parameter only works with the *randomwalk* SSS method.
**subsurface_type**

**Diffusion**

With a single layer, it can capture both surface detail and deep scattering. Designed to closely match the characteristics of the full Monte-Carlo simulation, while remaining an approximation that's much cheaper to evaluate than a full *Randomwalk*.

**Randomwalk (default)**

Unlike the empirical BSSRDF method based on diffusion theory, the *Randomwalk* method actually traces below the surface with a real random walk and makes no assumptions about the geometry being locally flat. This means it can take into account anisotropic scattering like brute-force volume rendering and produces much better results around concavities and small details. It can also be substantially faster for large scattering radius (i.e. large mean free path) compared to diffusion. On the other hand, *Randomwalk* can be slower in dense media (i.e. small mfp), does not support *sss_setname* for blending two surfaces together, may require redialing materials to achieve a similar look, and is more sensitive to non-closed meshes, "mouth bags", and internal geometry potentially casting shadows.

**Randomwalk v2**

This method scatters more accurately and deeply through highly-transparent/optically-thin objects, which produces SSS with more saturated colors around fine surface detail and heavily backlit regions of an object. Note that renders will be more costly and noisier than with the original method since randomwalks will be on average longer and more random.

**thin_walled**

*Thin Walled* can also provide the effect of a translucent object being lit from behind (the shading point is 'lit' by the specified fraction of the light hitting the reverse of the object at that point). It is recommended that this only be used with thin objects (single sided geometry) as objects with thickness may render incorrectly.
Thin Walled is ideal for thin (single sided) objects, like bubbles for example.

Thin-Walled Translucency
The appearance of this effect is like a thin sheet of paper letting some light through to the back side.

* For diffuse surfaces such as paper, enable Thin Walled and set Subsurface Weight to, for example, 0.5 to have half the light reflected and half transmitted.
In certain situations, Thin Walled may work fine with thickness. However, you should ensure that the Diffuse Ray Depth level is above 1 when using Thin Walled with objects that have thickness.

normal

Connect a Normal map here (usually exported from Mudbox or ZBrush).

Normal mapping works by replacing the interpolated surface normal by the one evaluated from an RGB texture, where each channel (Red, Green, Blue) correspond to the X, Y and Z coordinates of the surface normal. It can be faster than bump mapping since bump mapping requires evaluating the shader underneath at least three times.

tangent

The tangent map. Together with the shading normal, it defines the tangent coordinate system that the input vector applies to. If available from your sculpting tool, you should connect here the tangent map that the normal map relies on. If 0, the shader attempts the following actions to build the frame:

1. Look for vector user data named "tangent" and "bitangent."
2. Use the UV derivatives.
3. Build its own local frame.

The shader works in tangent space only. If your tangent map was exported in either world or object space, you could instead use the more generic space_transform shader.
coat

This attribute is used to coat the material. It acts as a clear-coat layer on top of all other shading effects. The coating is always reflective (with the given roughness) and is assumed to be dielectric. Examples would be the clear-coat layer for car paint or the sheen layer for a skin material. For example, for an extra oily layer or wet skin. Other examples would be objects that have been laminated or a protective film over an aluminum cell phone.

Coat can be used for materials such as metallic car paint, the reflective coating on a balloon or a bubble for example.
When combining a Coat (low roughness) on top of a Specular (high roughness), the sharp Coat disappears at the center due to Fresnel.

coat_color

This is the color of the coating layer's transparency.

In typical real-world cases, the Coat Color should always be white and is there mostly for artistic control.

In the example below, the red Coat Color acts as a clear coat layer, tinting the specular reflection underneath.
In the example below, a blue and white checker texture has been applied to the *Base Color* and the *Coat Color* has been changed to yellow. When combined, the yellow clear-coat layer on top of the base color gives a green tint similar to a semi-transparent layer of paint or lacquer.

![Base Color](image1) + ![Coat Color: Yellow](image2) = ![Coat and Base Color](image3)

You can also define how the coating affects the underlying *Base Color* by connecting a texture map to the *Coat Weight*.

![Checker connected to 'Coat Weight'](image4) + ![Yellow 'Coat Color' is now restricted to white areas of checker and blue 'Base Color' is unaffected](image5)

*Coat Color* can also be used to layer a stencil texture. In the example below, a type image has been connected to the *Coat Color*, and the *Coat Roughness* has been increased. Notice how it appears unaffected by the specular highlight that appears to 'sit' underneath it.
coat_roughness

Controls the glossiness of the specular reflections. The lower the value, the sharper the reflection. In the limit, a value of 0 will give you a perfectly sharp mirror reflection, while 1.0 will create reflections that are close to a diffuse reflection. You should connect a map here to get variation in the coat highlight.

0.1 (default) 0.3 0.6
coat_IOR

The IOR parameter (Index of Refraction) defines the material’s Fresnel reflectivity and is by default the angular function used. Effectively the IOR will define the balance between reflections on surfaces facing the viewer and on surface edges. You can see the reflection intensity remains unchanged, but the reflection intensity on the front side changes a lot.

coat_anisotropy

Anisotropy reflects and transmits light with a directional bias and causes materials to appear rougher or glossier in certain directions. The default value for Anisotropy is 0, which means 'isotropic.' As you move the control towards 1.0, the surface is made more anisotropic in the U axis.

Anisotropy is suitable for materials that have a clear brush direction such as brushed metal which has tiny grooves in which form a ‘stretched’ anisotropic reflection.
Many small discs form together to create an effect which is the anisotropic highlight.

Anisotropic reflections are suitable for brushed metal effects such as in the example below:

You may notice faceting appear in highlights when using Anisotropy. It is possible to remove the faceted appearance by enabling smooth subdivision tangents (via Arnold subdiv_smooth_derivs parameter). Take into account this requires a subdivision iteration of at least one in the polymesh to work.
Faceting seen in the anisotropic highlight


Increase Subdivision Iterations to remove anisotropic faceting.

cloth_rotation

The rotation value changes the orientation of the anisotropic reflectance in UV space. At 0.0, there is no rotation, while at 1.0 the effect is rotated by 180 degrees.

It is possible to assign textures to the rotation. When doing so, it is advisable to avoid texture filtering. This means disabling MIP-mapping and disabling the magnification filter, which by default is set to "smart bicubic." One way is to set the mipmap_bias of the image node to a strong negative value, like -8, which means "use 8 MIP levels higher resolution than usual."

cloth_normal

The Cloth Normal affects the Fresnel blending of the cloth over the base, so depending on the normal, the base will be more or less visible from particular angles. Uses for Cloth Normal could be a bumpy coat layer over a smoother base. This could include a rain effect, a carbon fiber shader or a car paint shader where you could use different normals (using e.g. flakes) for the coat layer and base layers.
Coat Normal is suited to situations where the layer may be uneven, for example, an oily or wet layer, rain on streets or glazing on food.

When you specify a Coat Normal, it affects just the coat, and none of the layers below it (diffuse, specular, transmission). In the example below right, the rock material appears to have a clear coat layer, when a bump texture is connected to the Coat Normal.
coat_affect_color

In the real world, when a material is coated there is a certain amount of internal reflections on the *inside* of the coating. This causes light to bounce onto the surface multiple times before escaping, allowing the material's color to have an enhanced effect. An example of this is varnished wood. This effect can be achieved by using *Coat Affect Color*. 
**coat_affect_roughness**

This causes the coating's roughness to have an effect on the underlying layer's roughness, simulating the blurring effect of being seen through the top layer.

In the example below, a checker texture has been connected to the *Coat Roughness* with *Metalness* set to 1.0. When set to 0, the *Coating* is unaffected, whereas at 1 the *Coating Roughness* is apparent.

![Coating examples](image)

**emission**

Controls the amount of emitted light. It can create noise, especially if the source of indirect illumination is very small (e.g. light bulb geometry).

![Emission examples](image)

0 (default) | 0.5 | 0.8
The emitted light color.

Increasing the number of *Diffuse samples* will help to reduce any noise in dark, indirectly lit areas of a scene when using emission.
opacity
Controls the degree to which light is not allowed to travel through it. Unlike Transmission, whereby the material still considers diffuse, specular, etc., opacity will affect the entire shader. Useful for retaining the shadow definition of an object, while making the object itself invisible to the camera.

You must ensure that Opaque is disabled for the mesh that the Standard Surface shader is assigned to when using Opacity.
**caustics**

This switch in the *Standard Surface* shader specifies whether specular or transmission bounces behind diffuse bounces are enabled or not. As caustics can be noisy, these are disabled by default.

To control noise from caustics, the global *Indirect Specular Blur* setting may be increased, which blurs out caustics to reduce noise at the cost of accuracy.

Arnold will require a high number of *Diffuse* samples to achieve a clean image when using caustics. Care should be taken when enabling this feature.

Notice that the areas in the scenes below are darker inside the glass. This is because the illumination inside the glass is essentially all caustics. Enabling caustics for the glass shader fixes this issue.
You may experience fireflies when using high Specular Weight, and low Specular Roughness with Caustics enabled. To reduce this type of noise, increase the Specular Roughness of the shader or Indirect Specular Blur globally.

| Disabled (default) | Enabled | Increasing the Specular Roughness reduces the noise |

Enabling caustics can add more believability when rendering eyes.
internal_reflections

Unchecking internal reflections will disable indirect specular and mirror perfect reflection computations when ray refraction depth is bigger than zero (when there has been at least one refraction ray traced in the current ray tree).

In the right image below, the sphere appears black because Internal Reflections is disabled in the Standard Surface shader assigned to the sphere.
**exit_to_background**

This will cause the *Standard Surface* shader to trace a ray against the background/environment when the maximum GI reflection/refraction depth is met and return the color that is visible in the background/environment in that direction. When the option is disabled, the path is terminated instead and returns black when the maximum depth is reached.

**indirect_diffuse**

The amount of diffuse light received from indirect sources only.
**indirect_specular**

The amount of specularity received from indirect sources only. Values other than 1.0 will cause the materials to not preserve energy, and global illumination may not converge.

**sss_set_name**

It is possible to tag multiple objects as belonging to the same SSS 'set' so that illumination will blur across object boundaries. A common use-case might be blurring between teeth and gum geometry. It is enabled by adding the constant STRING userdata sss_setname to the same value on the objects in the set.

This currently does not work with the randomwalk SSS method.
anisotropy_tangent

Lets you specify a custom tangent for specular anisotropic shading. For example, if the mesh has tangent user data, that could be linked here.

thin_film_thickness

Defines the actual thickness of the film between the specified min and max thickness (0 to 2000 (soft min/max)). This affects the specular, transmission and coat components. Normally this would be something like a noise map to give some variation to the interference effect. If the thickness becomes large like 3000 [nm], the iridescence effect will disappear, which is a physically correct behavior.

thin_film_IOR

The refractive index of the medium surrounding the material. Normally this is set to 1.0 for air.

The IOR for water is 1.33 and soap is about 1.5. Therefore the IOR for a soap bubble should be in the 1.33-1.4 range.
aov_id(1-8)

The color used to be output in AOV id (1-8). Connect a shader or texture here and it will be saved to the similarly named id AOV. AOV ids are useful for creating mattes to be used in compositing.

sheen

An energy-conserving sheen layer that can be used to approximate microfiber, cloth-like surfaces such as velvet and satin of varying roughness. Sheen is layered onto the diffuse component and its weight is determined with this attribute. Sheen can be thought of as the density or the combination of the density and length of the fibers.

As well as fabric, Sheen can be used to simulate leaves, fruit or the peach fuzz on a face (best when viewed from a distance).

sheen_color

The color of the fibers. Tints the color of the sheen contribution.
sheen_roughness

Modulates how much the microfibers diverge from the surface normal direction.

Sheen Roughness randomizes the fiber direction.