**基于物理的渲染材质验证器**

基于物理的渲染材质验证器 (Physically Based Rendering Material Validator) 是 Scene 视图中的一种绘制模式，它可以确保材质使用的值处于基于物理的着色器的建议参考值范围内。如果特定材质中的像素值落在参考范围之外，则材质验证器会突出显示不同颜色的像素以指示失败状态。

要使用材质验证器，请选择 Scene 视图的\_\_绘制模式 (Draw Mode)\_\_ 下拉菜单，默认情况下该菜单通常设置为 **Shaded**。

Scene 视图的绘制模式下拉菜单Scene 视图的绘制模式下拉菜单

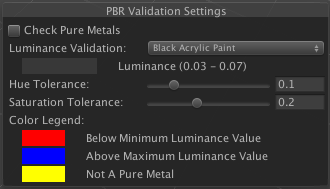
导航到 **Material Validation** 部分。材质验证器有两种模式：\_\_Validate Albedo\_\_ 和 **Validate Metal Specular**。

Scene 视图绘制模式下拉菜单中的 Material Validation 选项Scene 视图绘制模式下拉菜单中的 Material Validation 选项

**注意**：还可以使用 [Unity 的材质图表](file:///E:\UnityDocumentation\Manual\StandardShaderMaterialCharts.html)来检查建议值。在创作材质时，您仍需要使用这些图表来决定\_\_反照率\_\_和\_\_金属镜面反射\_\_值。但是，材质验证器提供了一种可视化的编辑器内部验证方式，可让您在资源处于场景中时快速检查材质的值是否有效。

**另请注意**：验证器仅适用于线性颜色空间。基于物理的渲染不适用于伽马颜色空间，因此如果您要使用基于物理的渲染 (PBR) 和 PBR 材质验证器，那么同时还应使用[线性颜色空间](file:///E:\\UnityDocumentation\\Manual\\LinearRendering-LinearOrGammaWorkflow.html)。

**Validate Albedo 模式**

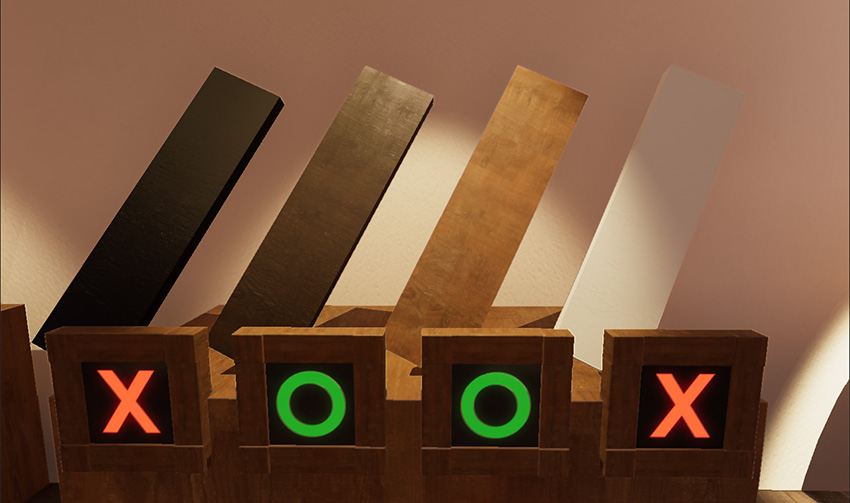
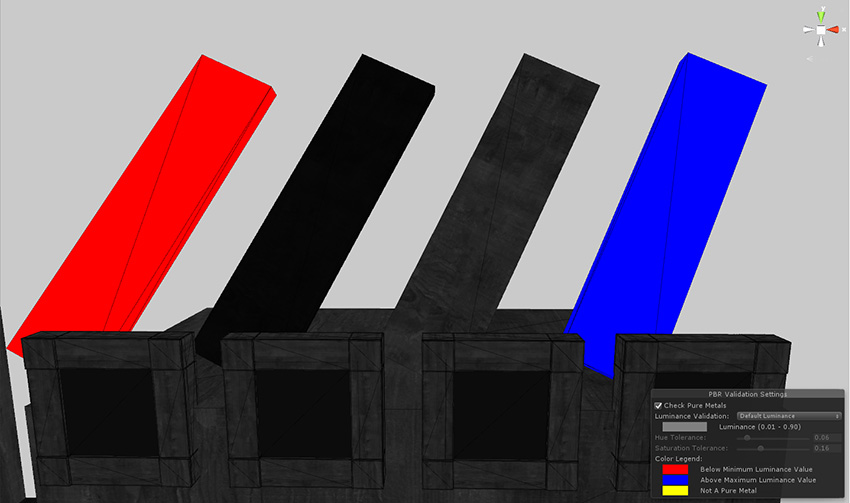
处于 Validate Albedo 模式时在 Scene 视图中显示的 PBR 验证设置 (PBR Validation Settings)

将 Material Validation 设置为 Validate Albedo 时，在 Scene 视图中显示的 PBR 验证设置 (PBR Validation Settings) 如下。

| **属性：** | | **功能：** |
| --- | --- | --- |
| **Check Pure Metals** | | 如果希望材质验证器以黄色突出显示所发现的由 Unity 定义为金属性但具有非零反照率值的像素，请启用此复选框。有关更多详细信息，请参阅下面的[纯金属](file:///E:\\UnityDocumentation\\Manual\\MaterialValidator.html" \l "pureMetals)。默认情况下未启用此选项。 |
| **Luminance Validation** | | 使用下拉选单选择材质验证器的预设配置。如果选择 **Default Luminance** 以外的任何选项，还可以调整 Hue Tolerance 和 Saturation Tolerance。此属性名称下方的色带表示该配置的反照率颜色。下拉选单下方的 Luminance 值表示最小和最大亮度值。材质验证器会突出显示亮度值超出这些值的所有像素。默认情况下，此选项设置为 **Default Luminance**。 |
| **Hue Tolerance** | | 检查材质的反照率颜色值时，此滑动条允许您控制材质色调与验证配置中的色调之间允许的误差量。 |
| Saturation Tolerance | | 检查材质的反照率颜色值时，此滑动条允许您控制材质饱和度与验证配置中的饱和度之间允许的误差量。 |
| **Color Legend** | | 这些颜色对应于材质验证器在材质像素超出定义值时将会显示在 Scene 视图中的颜色。 |
|  | **Red** Below Minimum Luminance Value | 材质验证器以红色突出显示低于 **Luminance Validation** 中定义的最小亮度值的任何像素（表示它们太暗）。 |
|  | **Blue** Above Maximum Luminance Value | 材质验证器以蓝色突出显示高于 **Luminance Validation** 中定义的最大亮度值的任何像素（表示它们太亮）。 |
|  | **Yellow** Not A Pure Metal | 如果启用了 Check Pure Metals 选项，则材质验证器会以黄色突出显示由 Unity 定义为金属性但具有非零反照率值的像素。有关更多详细信息，请参阅下面的“纯金属”。 |

Unity 的[材质图表](file:///E:\\UnityDocumentation\\Manual\\StandardShaderMaterialCharts.html)定义了非金属的标准亮度范围为 50–243 sRGB，而金属的标准亮度范围为 186–255 sRGB。**Validate Albedo** 模式会使用不同颜色显示处于这些范围之外的任何像素，以指示该值太低或太高。

在下面的示例中，第一个纹理低于最小亮度值，因此太暗。第四个纹理高于最小亮度值，因此太亮。

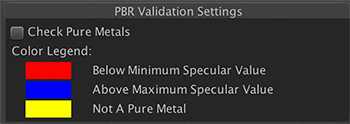
一个场景（未启用材质验证器）中的第一个和第四个材质具有不正确的反照率值 启用了材质验证器并设置为 Validate Albedo 模式的相同场景。低于最小亮度值的纹理为红色。高于最小亮度值的纹理为蓝色

[材质图表](file:///E:\UnityDocumentation\Manual\StandardShaderMaterialCharts.html)提供常见材质的反照率值。反照率值的亮度对产生的漫射光量有巨大影响，因此[全局光照](file:///E:\UnityDocumentation\Manual\GIIntro.html)烘焙必须确保不同的材质类型位于正确的亮度范围内（彼此成比例），这一点十分重要。为了方便正确设置这些值，可从 Luminance Validation 下拉选单的预设选项中进行选择，该下拉选单提供了常用的材质反照率值以便验证特定材质类型的亮度范围。

**覆盖默认亮度值**

根据项目的美术风格，您可能希望材质的亮度值与预设的亮度范围不同。在这种情况下，可使用自己的值覆盖材质验证器使用的内置反照率值。要覆盖预设亮度范围，请将每种所需材质类型的 [AlbedoSwatchInfo](file:///E:\UnityDocumentation\ScriptReference\Rendering.AlbedoSwatchInfo.html) 值数组分配给属性 [EditorGraphicsSettings.albedoSwatches](file:///E:\UnityDocumentation\ScriptReference\Rendering.EditorGraphicsSettings-albedoSwatches.html)。

**Validate Metal Specular 模式**

处于 Validate Metal Specular 模式时的 PBR 验证设置 (PBR Validation Settings)

将 **Material Validation** 设置为 **Validate Metal Specular** 时，在 Scene 视图中显示的 PBR 验证设置 (PBR Validation Settings) 如下。

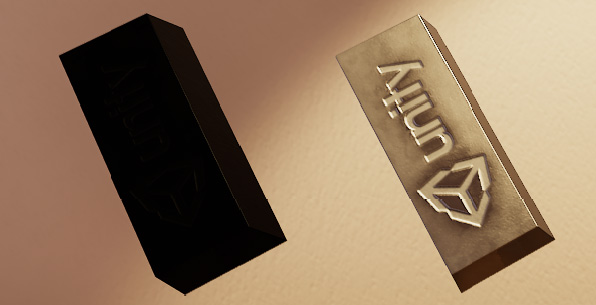
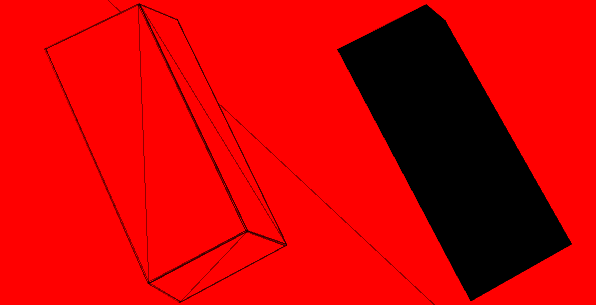
| ***属性：*** | | ***功能：*** |
| --- | --- | --- |
| **Check Pure Metals** | | 如果希望材质验证器以黄色突出显示所发现的由 Unity 定义为金属性但具有非零反照率值的像素，请启用此复选框。有关更多详细信息，请参阅下面的“纯金属”。默认情况下未启用此选项。 |
| **Color Legend** | | 这些颜色对应于材质验证器在材质像素无效时（这意味着它们的镜面反射值超出了该材质类型（金属性或非金属性）的有效范围）在 Scene 视图中显示的颜色。有效范围见下表。 |
|  | **Blue** Below Minimum Specular Value | 材质验证器以红色突出显示低于最小镜面反射值的所有像素。（40 表示非金属性，155 表示金属性）。 |
|  | **Red** Above Maximum Specular Value | 材质验证器以蓝色突出显示高于最大镜面反射值的所有像素。（75 表示非金属性，255 表示金属性）。 |
|  | **Yellow** Not A Pure Metal | 如果启用了 **Check Pure Metals** 选项，则材质验证器会以黄色突出显示由 Unity 定义为金属性但具有非零反照率值的像素。有关更多详细信息，请参阅下面的“纯金属”。 |

Unity 的[材质图表](file:///E:\\UnityDocumentation\\Manual\\StandardShaderMaterialCharts.html)定义了两个独立的镜面反射颜色范围：

* \_\_非金属性材质\_\_：40–75 sRGB
* \_\_金属性材质\_\_：155 - 255 sRGB

在 Unity 中，所有非金属性材质都具有始终处于正确范围内的恒定镜面反射颜色。然而，金属性材质通常具有太低的镜面反射值。为了帮助您识别存在此问题的金属性材质，材质验证器的 **Validate Metal Specular** 模式使用颜色来标示镜面反射颜色值太低的所有像素。根据定义，这包括所有非金属性材质。

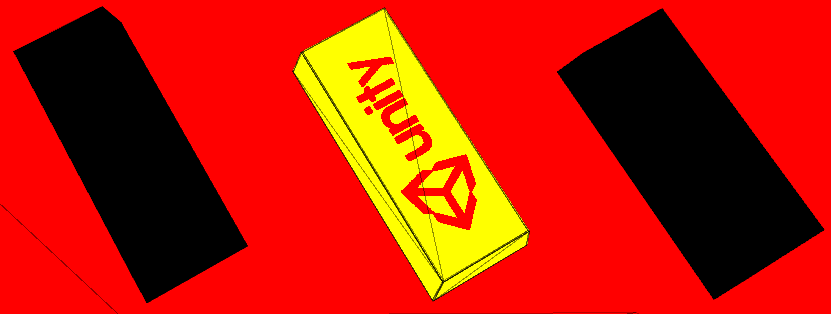
在下面的示例中，左侧材质低于最小镜面反射值，因此太暗。这也适用于场景的背景。右侧材质的镜面反射值处于有效范围内。

包含两种金属性材质的场景。左侧材质的金属性镜面反射值不正确 启用了材质验证器并设置为 **Validate Metal Specular** 模式的相同场景

**纯金属**

Unity 将镜面反射颜色大于 155 sRGB 的基于物理的着色材质定义为金属性材质。Unity 将金属性材质定义为\_\_纯金属\_\_

如果非金属性表面的镜面反射颜色值太高，但反照率值为非零，则通常是由于创作错误导致的。材质验证器还有一个名为 **Check Pure Metals** 的选项。启用此选项后，材质验证器会以黄色显示由 Unity 定义为金属性但具有非零反照率值的所有材质。在下面的图像中可以看到这方面的一个例子。此处显示了三种材质，左侧和右侧材质为纯金属，但中间的材质不是，因此材质验证器将其显示为黄色：

包含三种金属性材质的场景。中间的材质不是纯金属（它具有非零反照率值） 启用了\_\_材质验证器**、设置为** Validate Metal Specular\_\_ 模式并启用了 Check Pure Metals 的相同场景

在上面的第二张图中，背景为红色，这是因为背景中的材质低于材质验证器的 **Validate Metal Specular** 模式下的最小镜面反射值。

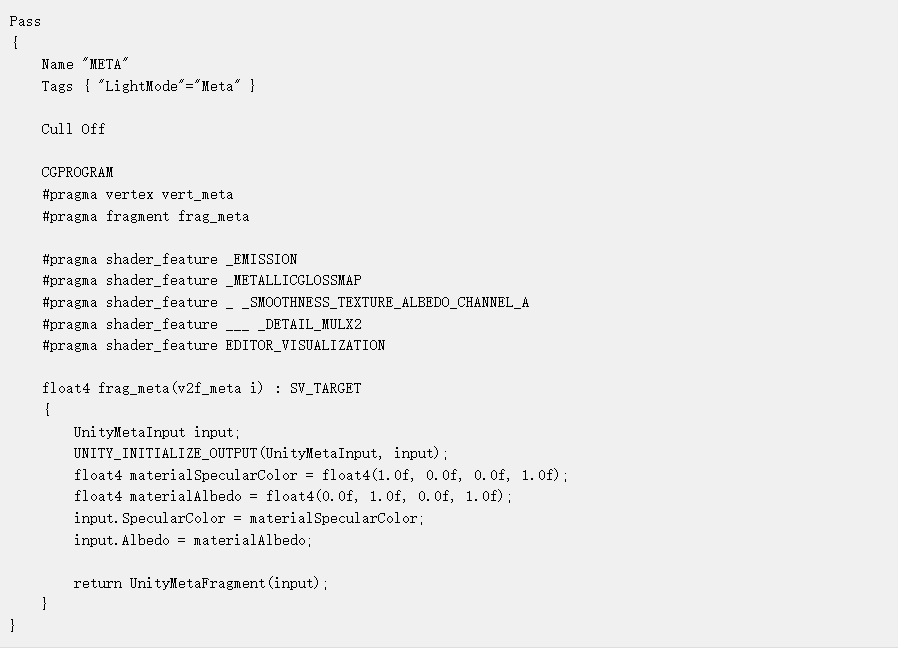
对于金属性和非金属性属性相结合的复杂材质，纯金属检查器可能会拾取一些无效像素，但如果材质完全无效，则通常表示存在创作错误。

**实现**

只要材质使用 Unity [标准着色器](file:///E:\UnityDocumentation\Manual\shader-StandardShader.html)或[表面着色器](file:///E:\UnityDocumentation\Manual\SL-SurfaceShaders.html)，即可采用材质验证器。但是，自定义着色器需要一个名为 "META" 的 pass。大多数支持光照贴图的自定义着色器已经定义了此 pass。请参阅有关 [Meta pass](file:///E:\UnityDocumentation\Manual\MetaPass.html) 的文档以了解更多详细信息。

执行以下步骤，使自定义着色器与材质验证器兼容：

1.将以下 pragma 添加到 Meta pass：#pragma shader\_feature EDITOR\_VISUALIZATION 2.在 UnityMetaInput 结构中，将材质的镜面反射颜色分配给名为 SpecularColor 的字段，如下面的代码示例所示。

以下是自定义 Meta pass 的示例：

# 通过脚本访问和修改材质参数

在检视面板中，您所看到的所有材质参数都可以通过脚本来访问，这可以允许您在运行时修改材质的行为，利用这一点您可以制作出基于材质属性的动画效果。

在游戏运行过程中，您可以动态修改材质的数值属性、更改颜色，以及更换纹理。在这种工作中最常用到的函数有这些：

| **函数名称** | **用途** |
| --- | --- |
| [SetColor](file:///E:\UnityDocumentation\ScriptReference\Material.SetColor.html) | 更改材质的颜色（例如反照率着色颜色） |
| [SetFloat](file:///E:\UnityDocumentation\ScriptReference\Material.SetFloat.html) | 设置浮点值（例如，法线贴图乘数） |
| [SetInt](file:///E:\UnityDocumentation\ScriptReference\Material.SetInt.html) | 在材质中设置整数值 |
| [SetTexture](file:///E:\UnityDocumentation\ScriptReference\Material.SetTexture.html) | 为材质分配新纹理 |

完整的材质处理函数列表可以参考 [Material 类脚本参考](file:///E:\UnityDocumentation\ScriptReference\Material.html)。

有一点需要注意的是，这些函数**只会设置当前着色器能够使用**的材质属性。这意味着，如果着色器不使用任何纹理，或者根本没有绑定任何着色器，则调用 [SetTexture](file:///E:\UnityDocumentation\ScriptReference\Material.SetTexture.html) 将无效。即使稍后设置了需要纹理的着色器，也是如此。因此，建议您在设置任何属性之前，先设置着色器。一旦完成设置后，您就可以在使用同一组纹理和属性值的着色器之间切换，所有设定的值都会得到保留。

这些函数的作用与所有简单着色器（如旧版着色器）以及除标准着色器以外的内置着色器（如粒子、精灵、UI 和无光照着色器）相同。但是，对于使用标准着色器的材质，在完全修改材质之前，您还必须了解一些其他要求。

## 使用标准着色器编写脚本的特殊要求

如果要在运行时修改材质，标准着色器有一些额外的要求，因为标准着色器在幕后实际上是由许多不同着色器合为一体的着色器。

这些不同类型的着色器称为[着色器变体](file:///E:\\UnityDocumentation\\Manual\\SL-MultipleProgramVariants.html)，可以把它考虑为所有着色器功能的组合，其中部分功能处于激活状态，其余则处于未激活状态。

例如，如果您为材质设置了一个[法线贴图](file:///E:\\UnityDocumentation\\Manual\\StandardShaderMaterialParameterNormalMap.html)，就会激活支持法线贴图的着色器变体。如果随后又设置了[高度贴图](file:///E:\UnityDocumentation\Manual\StandardShaderMaterialParameterHeightMap.html)，就会激活同时支持法线贴图和高度贴图的着色器变体。

这是一种很好的机制，它意味着当您使用标准着色器时，如果某个材质中没有使用法线贴图，它就不会存在法线贴图的性能开销，因为您实际运行的着色器省略了这部分功能的代码。这也意味着，如果您从不使用某个功能组合（例如“高度贴图”(HeightMap) 和“发光”(Emissive) 的组合），则您的构建版本中将完全忽略掉该变体；实际上，您通常只会用到标准着色器中极少数的可能变体。

Unity 会阻止简单地在构建中包含所有可能的着色器变体的行为，因为这将是一个非常大的数字，可能有数万个！这个巨大的数字不仅因为材质检视面板中每种可能的功能组合方式极其繁多，还因为针对不同渲染方案（例如是否使用 HDR、光照贴图、GI、雾效等）的每种功能组合存在多种变体。囊括所有这些可能性将导致加载缓慢、内存消耗高，并增加构建文件尺寸和构建时间。

所以 Unity 的替代方案是：通过检查项目中使用的材质资源来追踪您使用过的着色器变体。无论在项目中加入了哪些标准着色器变体，这些变体都会包含在构建中。

这个方案在通过脚本访问标准着色器材质的情况下，存在两个问题。

### 1.您必须为所需的标准着色器变体启用正确的关键字

如果使用脚本来更改材质，而这种更改会导致材质使用标准着色器的另一种不同变体，则必须**使用** [**EnableKeyword**](file:///E:\UnityDocumentation\ScriptReference\Material.EnableKeyword.html) **函数启用该变体**。如果您开始使用一个并未被材质使用过的着色器功能，另一个额外的变体就会被启用起来。例如将法线贴图分配给先前没有此贴图的材质，或者将初始为零的发光级别设置为大于零的值。

用来启用标准着色器功能的关键字如下：

| **关键字** | **功能** |
| --- | --- |
| \_NORMALMAP | [法线贴图](file:///E:\UnityDocumentation\Manual\StandardShaderMaterialParameterNormalMap.html) |
| \_ALPHATEST\_ON | [“镂空”透明度](file:///E:\UnityDocumentation\Manual\StandardShaderMaterialParameterRenderingMode.html)渲染模式 |
| \_ALPHABLEND\_ON | [“淡化”透明度](file:///E:\UnityDocumentation\Manual\StandardShaderMaterialParameterRenderingMode.html)渲染模式 |
| \_ALPHAPREMULTIPLY\_ON | [“透明”透明度](file:///E:\UnityDocumentation\Manual\StandardShaderMaterialParameterRenderingMode.html)渲染模式 |
| \_EMISSION | [发射颜色](file:///E:\UnityDocumentation\Manual\StandardShaderMaterialParameterEmission.html)或[发射贴图](file:///E:\UnityDocumentation\Manual\StandardShaderMaterialParameterEmission.html) |
| \_PARALLAXMAP | [高度贴图](file:///E:\UnityDocumentation\Manual\StandardShaderMaterialParameterHeightMap.html) |
| \_DETAIL\_MULX2 | [辅助“细节”贴图](file:///E:\UnityDocumentation\Manual\StandardShaderMaterialParameterDetail.html)（反照率和法线贴图） |
| \_METALLICGLOSSMAP | 金属性工作流程中的[金属性/平滑度贴图](file:///E:\UnityDocumentation\Manual\StandardShaderMaterialParameterMetallic.html) |
| \_SPECGLOSSMAP | 镜面反射工作流程中的[镜面反射/平滑度贴图](file:///E:\UnityDocumentation\Manual\StandardShaderMaterialParameterSpecular.html) |

使用上面的关键字足以让您的材质修改脚本在编辑器运行模式中起效。

但是，由于 Unity 仅通过检查项目中使用的材质来确定要在构建中包含哪些变体，因此它不会包含仅在运行时通过脚本使用到的变体。

这就是说，如果您在脚本中为材质启用 \_PARALLAXMAP 关键字，但是项目中未使用与该功能组合匹配的材质，则视差贴图将无法在最终构建中生效（即使在编辑器中看起来正常）。这是因为构建过程将会忽略该变体，因为它看起来不是必需的。

### 2.您必须确保 Unity 在构建中包含所需的着色器变体

为此，您需要确保 Unity 知道您要使用该着色器变体，具体方法就是在资源中包含至少一个该类型的材质。该材质**必须应用在场景中**，或者也可以将其放入 [Resources 文件夹](file:///E:\UnityDocumentation\Manual\LoadingResourcesatRuntime.html)，否则该材质处于未使用状态，Unity 仍会在构建中忽略它。

通过完成上述两个步骤，即可在运行时使用标准着色器修改材质。

如果您有兴趣了解有关着色器变体的详细信息以及如何构建自己的变体，请阅读[制作多个着色器程序变体的信息](file:///E:\UnityDocumentation\Manual\SL-MultipleProgramVariants.html)。