MatteID in Maya

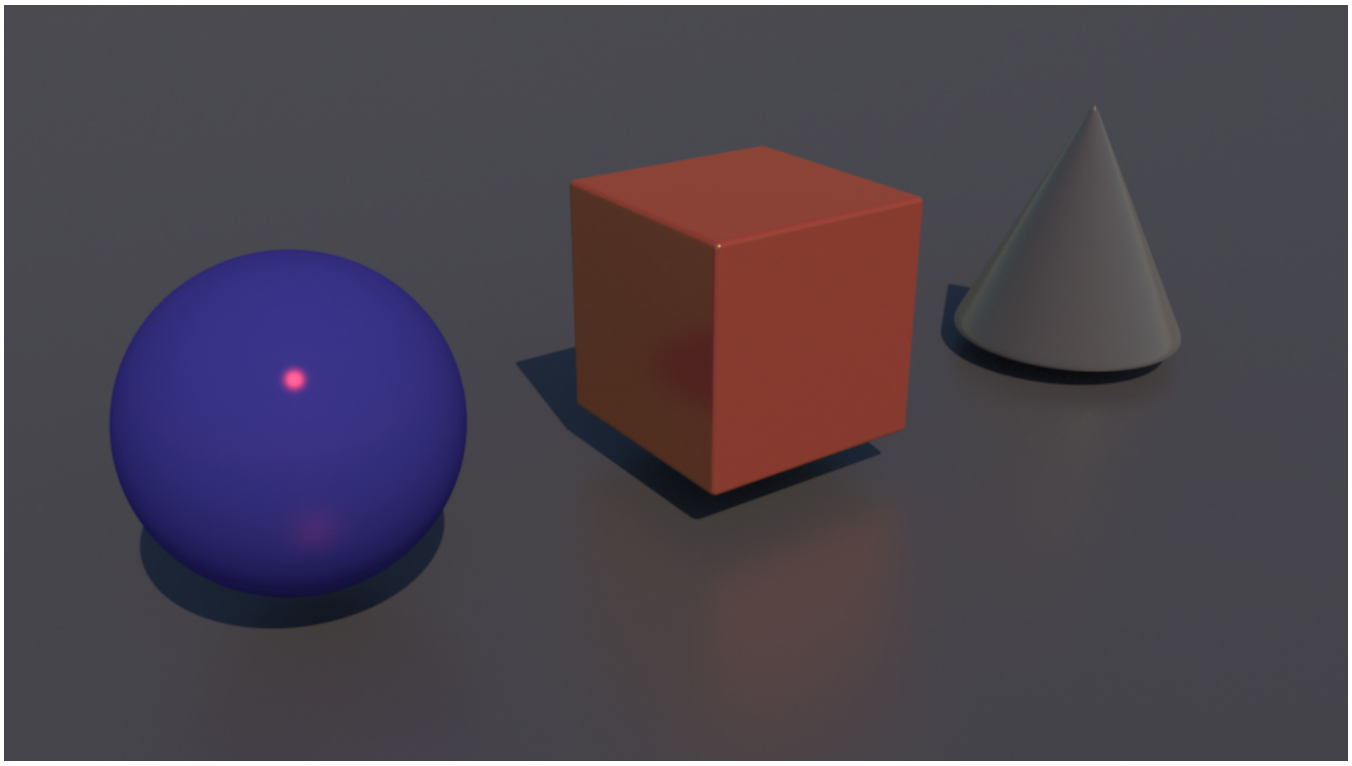
This workflow is deprecated.

Have you tried Cryptomatte instead of MatteIDs for automatic ID generation?

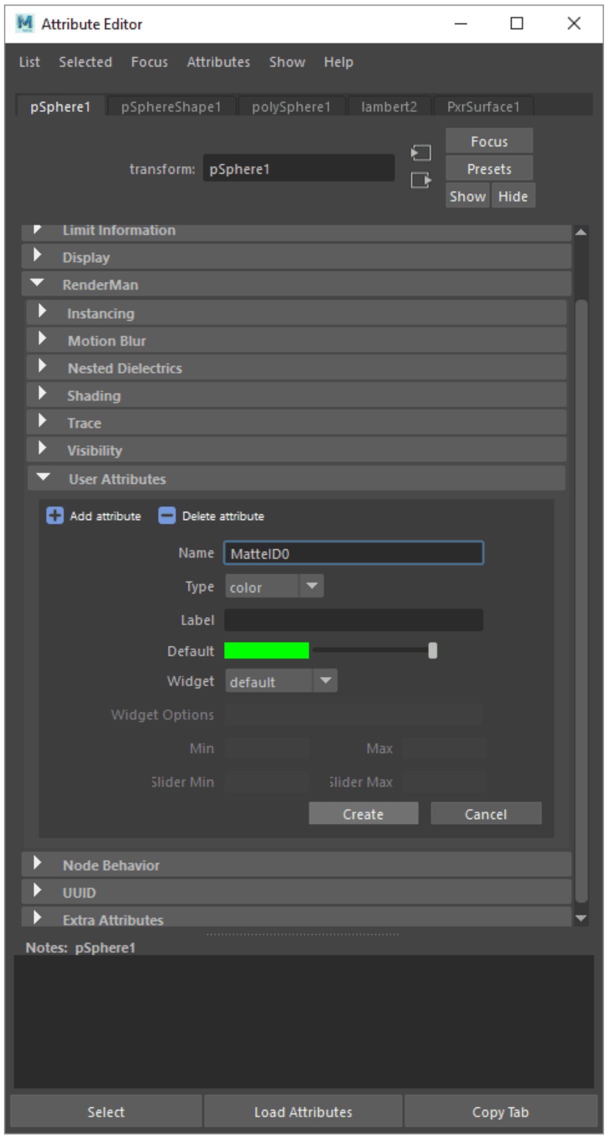
Assigning Matte IDs to rendered object and outputting them as part of the AOVs can be useful for later compositing because this allows compositors to isolate objects.

In the most basic way, you can do this in a few steps. We describe this below for material IDs but the process is similar for shapes.

Below we have a simple example with three shapes. This can be used for MatteID0 since a color output has three easy possibilities -- Red, Green, and Blue. (We will cover using patterns after this.)



First we'll assign a Green MatteID to the sphere. We can do this by selecting the transform and adding the RenderMan user attribute for MatteID0:



transform: pSphere1

Name: MatteID0

Type: color

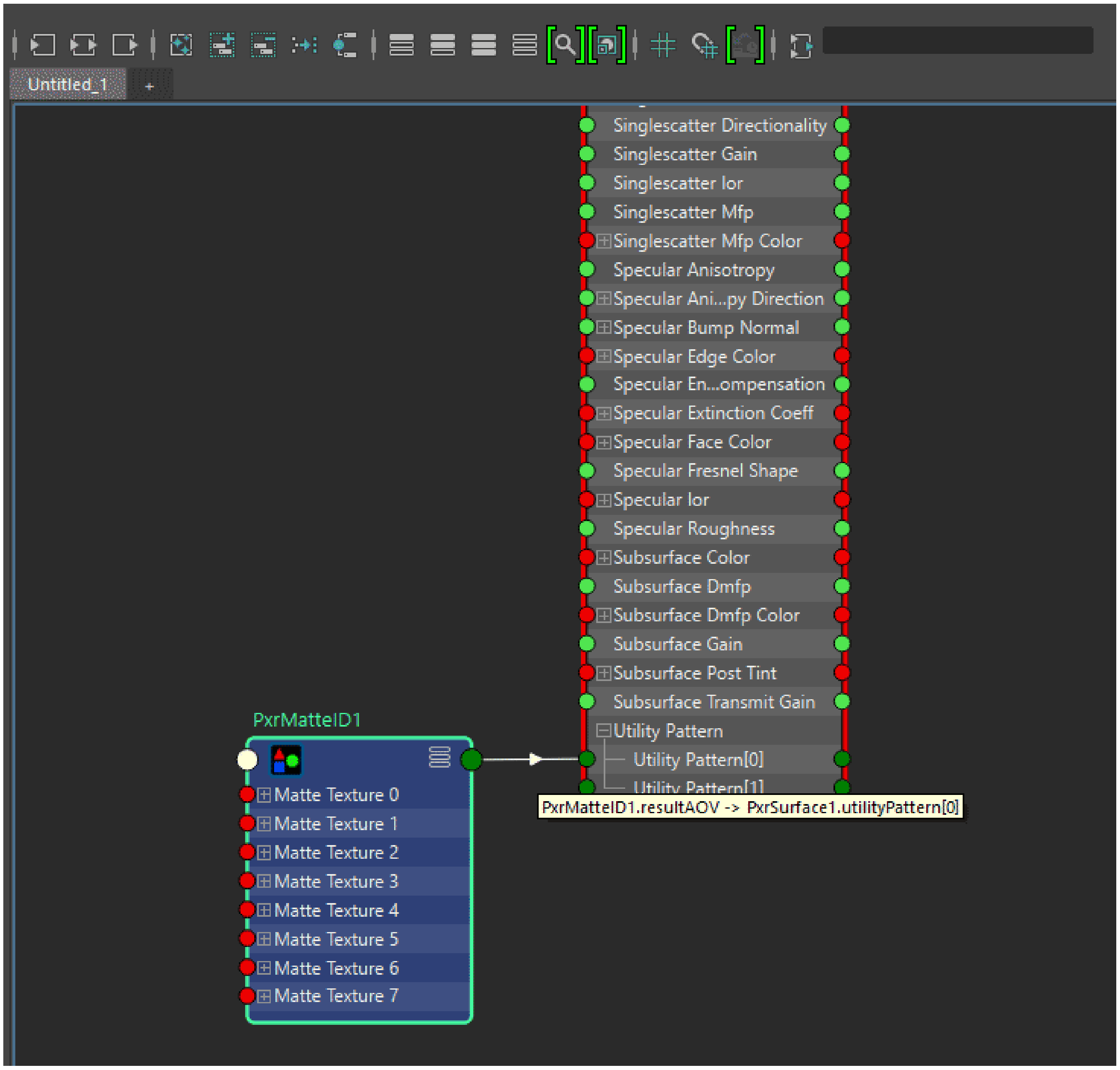
Label:

Default: 🟩

Widget: default

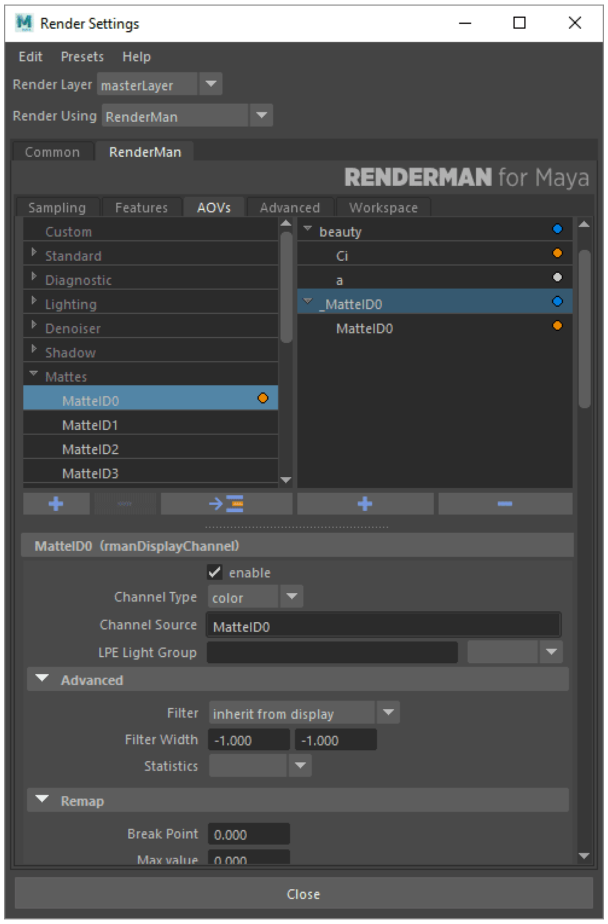
Notes: pSphere1

If you were to render at this point, even if you added the correct render pass, it would not work. You need to connect a PxrMatteID to the material. You can do this using the Node Editor and connecting to the Utility Pattern Connection of the PxrSurface material:



PxrMatteID1.resultAOV -> PxrSurface1.utilityPattern[0]

Now add the appropriate pass to the render outputs:



Render Layer masterLayer

Render Using RenderMan

MatteID0 (rmanDisplayChannel)

Channel Type color

Channel Source MatteID0

LPE Light Group

Filter inherit from display

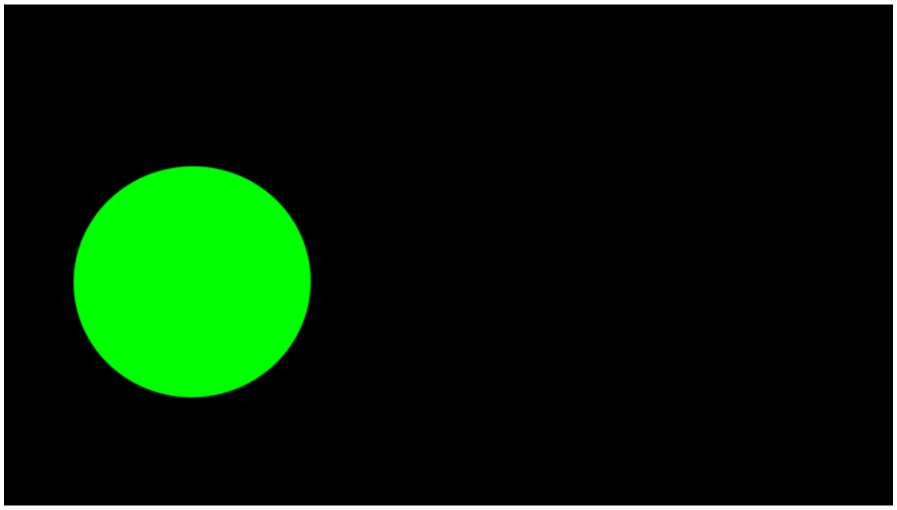
Filter Width -1.000 -1.000

Statistics

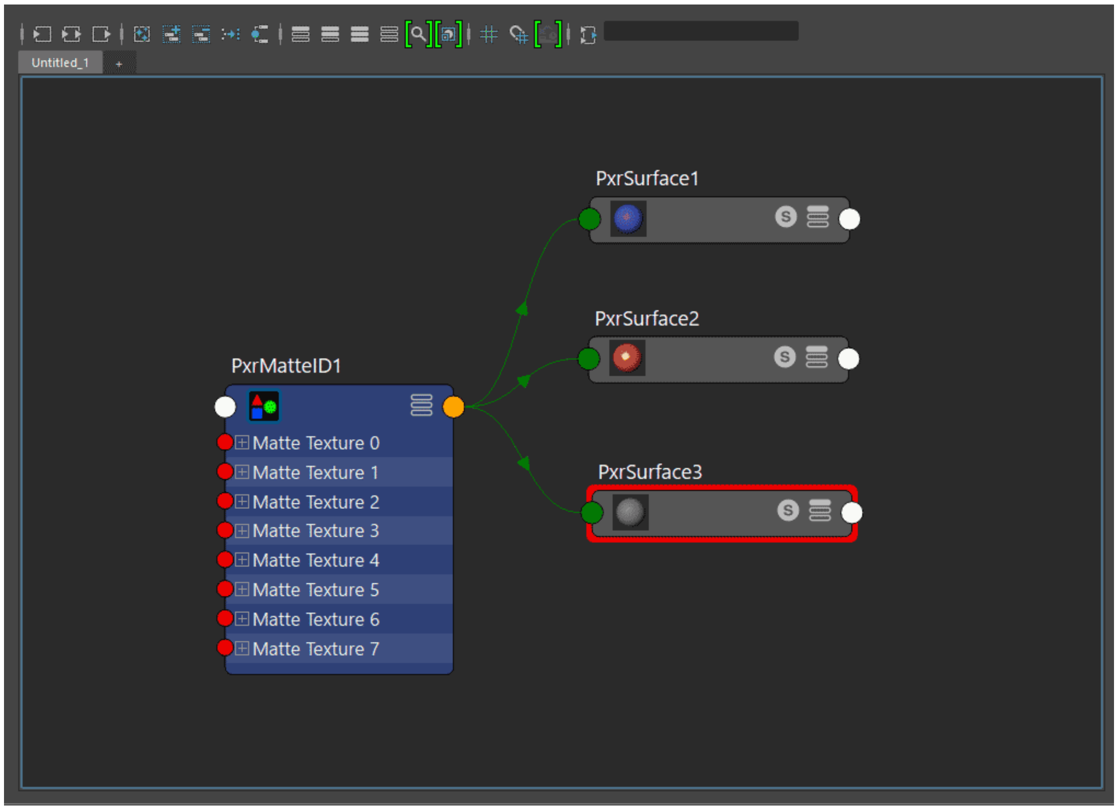
Break Point 0.000

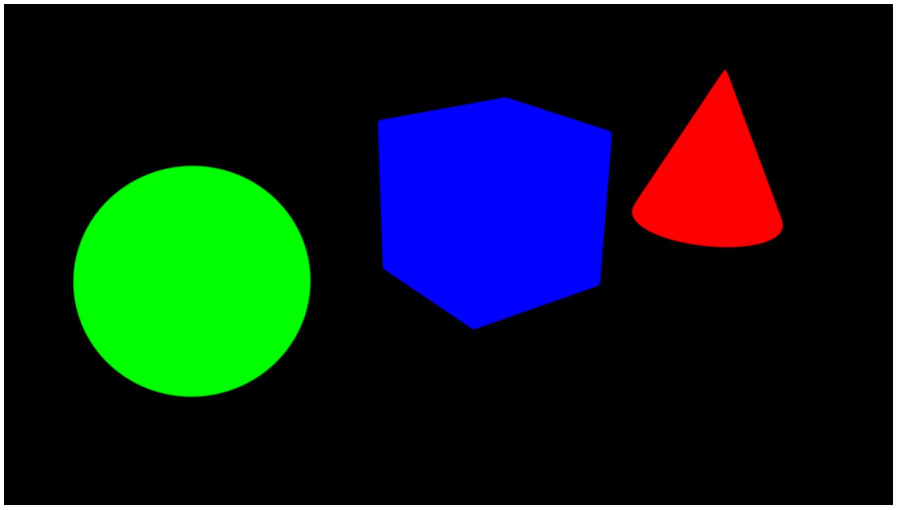
Max value 0.000

The resulting MattID0 pass looks like the one below:



You can repeat this process for the other materials, adding one Blue and a Red. You can connect the same PxrMatteID to the other PxrSurface Utility Pattern connections as shown below:

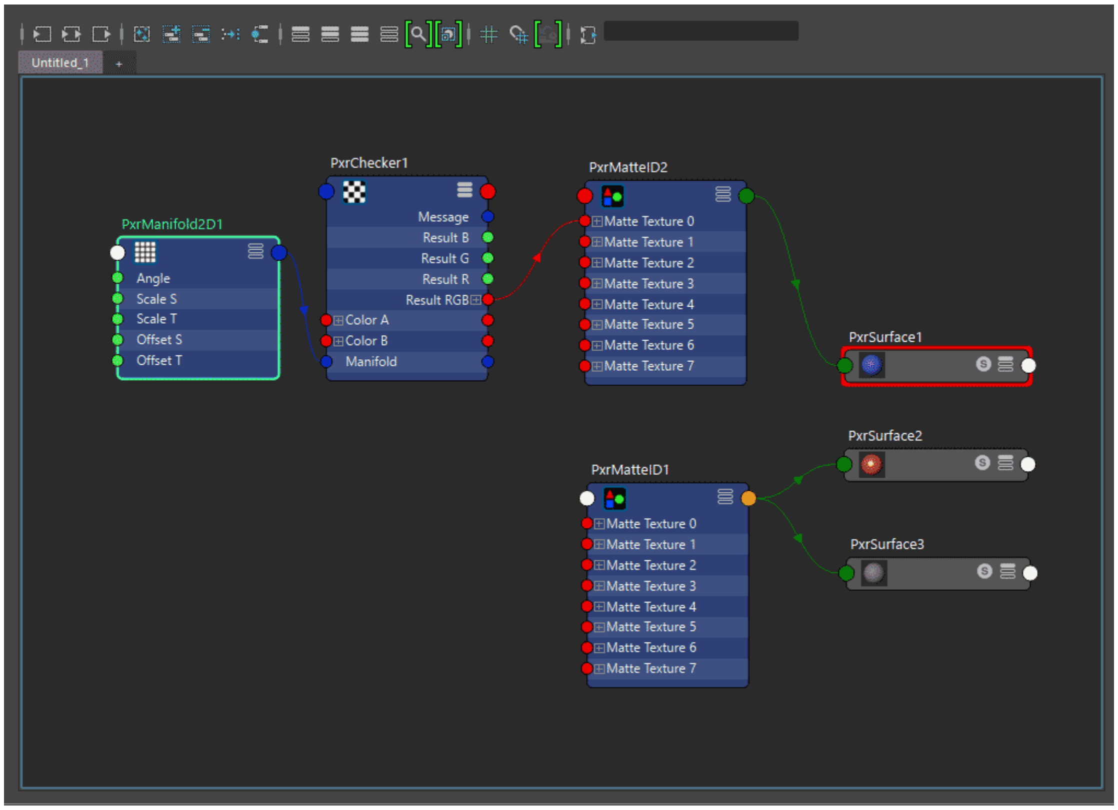


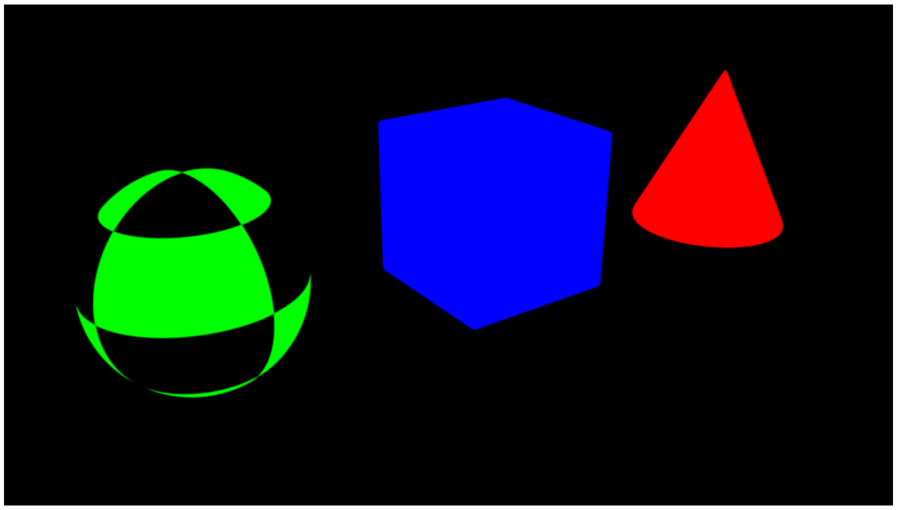


This works fine, but what if you needed to pass through a pattern as your result?

If you recall, we didn't make any changes to the PxrMatteID node. It was left at default. We will leave this one alone but disconnect it from PxrSurface1 assigned to the sphere. We're doing this so we can affect just one object and not all three.

Create a new PxrMatteID and connect it to the PxrSurface1 for the sphere. In the MatteID0 connection connect a PxrChecker and a manifold as shown below and then render.

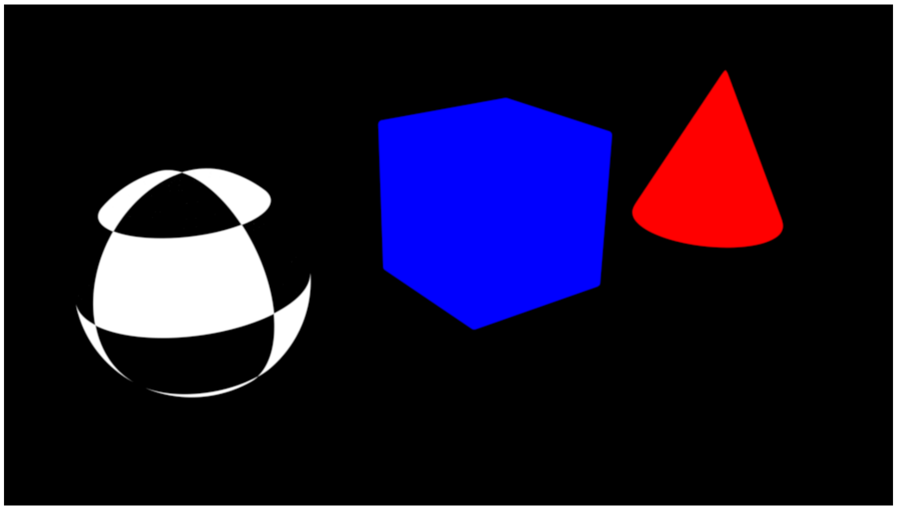




Ok, that sort-of worked, but it's green.

This is because in the material we still have the color set to green in the PxrSurfaceMaterial.

This color is multiplied against the pattern we connect to the PxrMatteID. This can be useful, but maybe we want a black and white checker this time. Set the color to white (since it's being multiplied, this should pass the pattern through unchanged.)



Now you can add additional PxrMatteIDs and chain together more patterns for different results in your renders. Your compositor will thank you.