Yel Scrypt Fire Studio

Cyberpunk Melee Weapons Pack URP Upgrade

BUILT-IN - UNIVERSAL RENDER PIPELINE SWITCHING
SHADER GRAPH ASSET DOCUMENTATION

Version 2.1

CONTENT

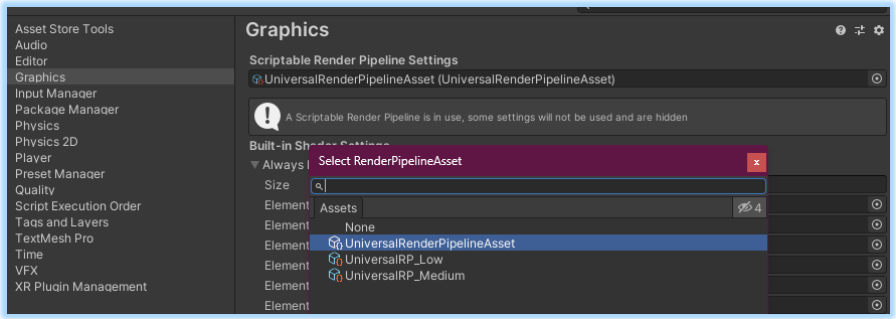
BUILT-IN – UNIVERSAL RENDER PIPELINE SWITCHING

This package uses standard materials. These materials can be switched between Built-in and URP render pipelines.

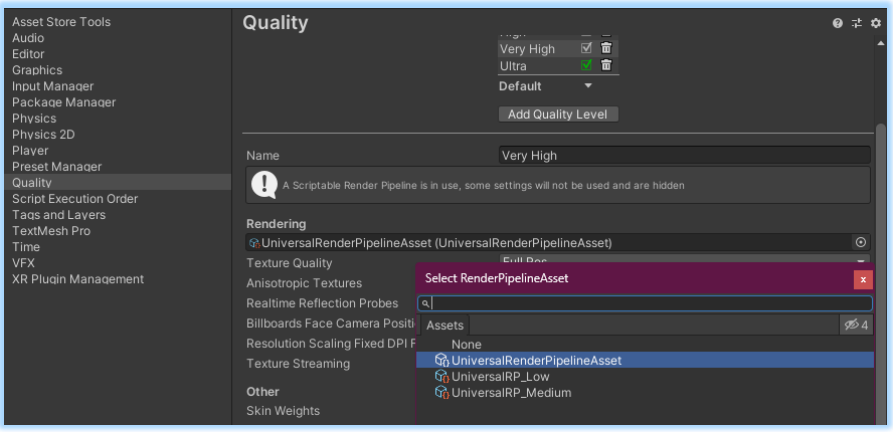
URP ASSETS SETUP

*This step is not needed if your project is configured in URP by default. *

- Go to **Edit > Project Settings**
- In the configuration windows, go to **Graphics** and set the URP asset, clicking on the circle and selecting the asset.



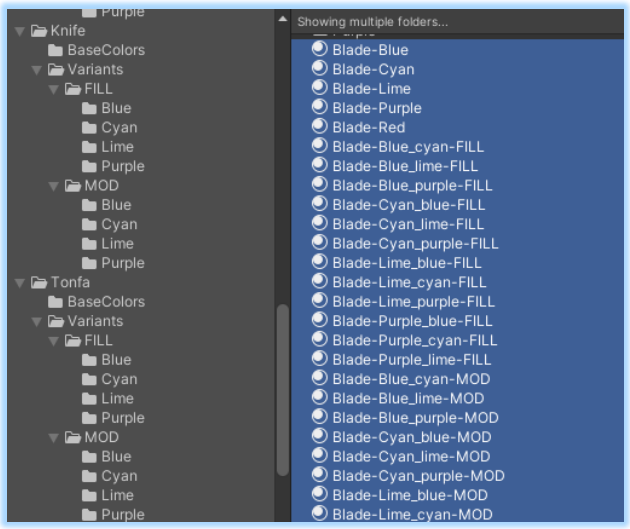
- Go to **Quality** and set the same URP asset.



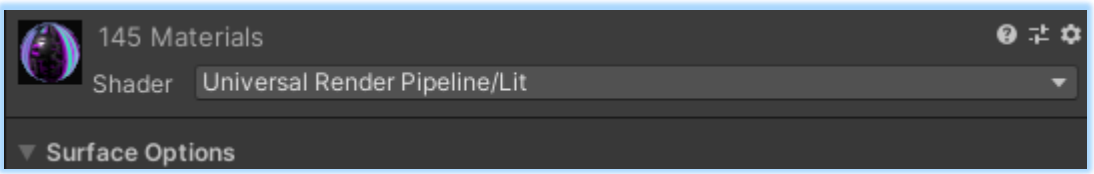
- **If files are not recognized, please reimport the URP assets.**
- To change from URP to Bult-in, just select **None** instead of the URP asset.

FROM URP

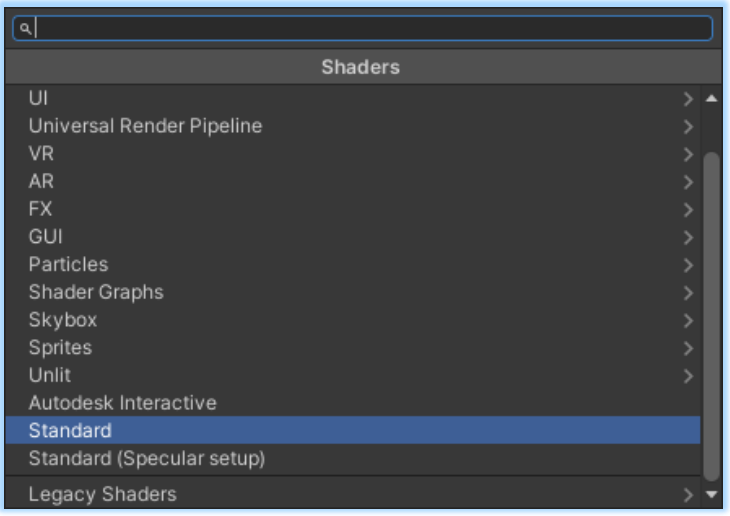
- Select all the material will be switched (must be set in the same “Lit” shader).



- Go to the shader selector in the Inspector.



- Select **Standard** from the list of shaders.

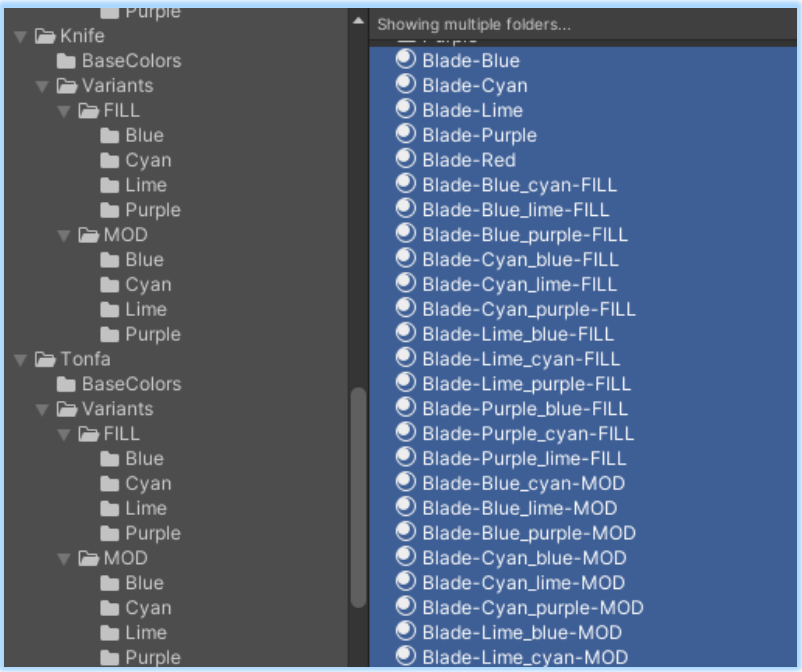


- Now, materials are in **Built-In Standard** shader.

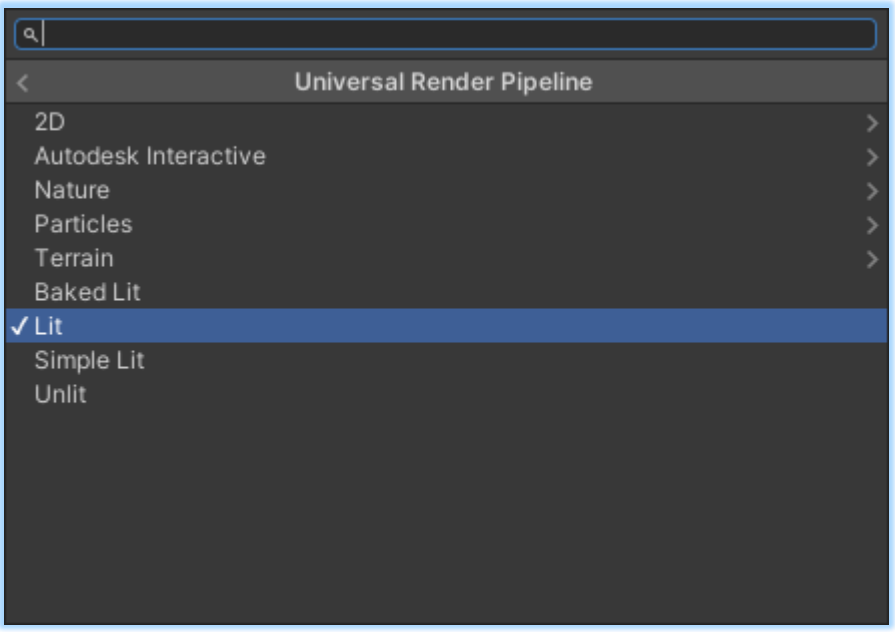
FROM BUILT-IN

METHOD #1

- Select all the material will be switched.



- Go to the shader selector in the Inspector.
- Select **Universal Render pipeline>Lit** from the list of shaders.

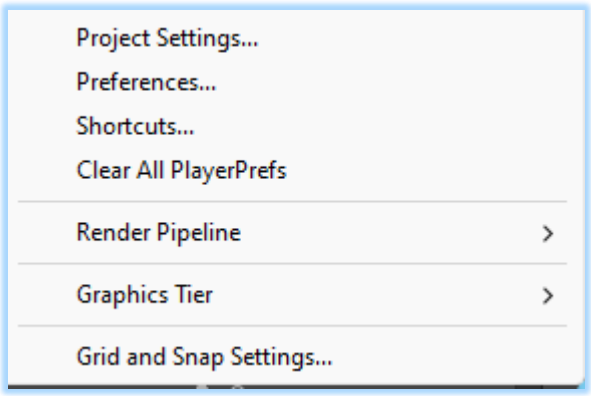


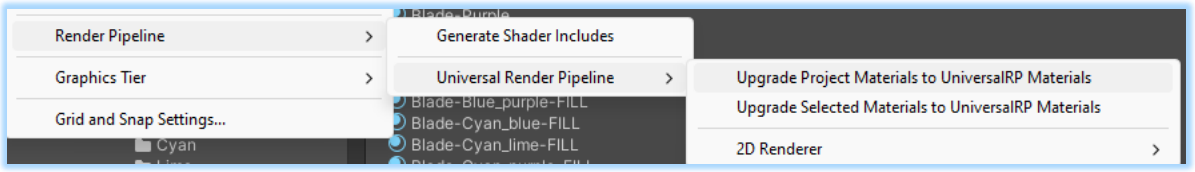
- All materials will be switched to URP.

METHOD #2

- Select all the material will be switched.

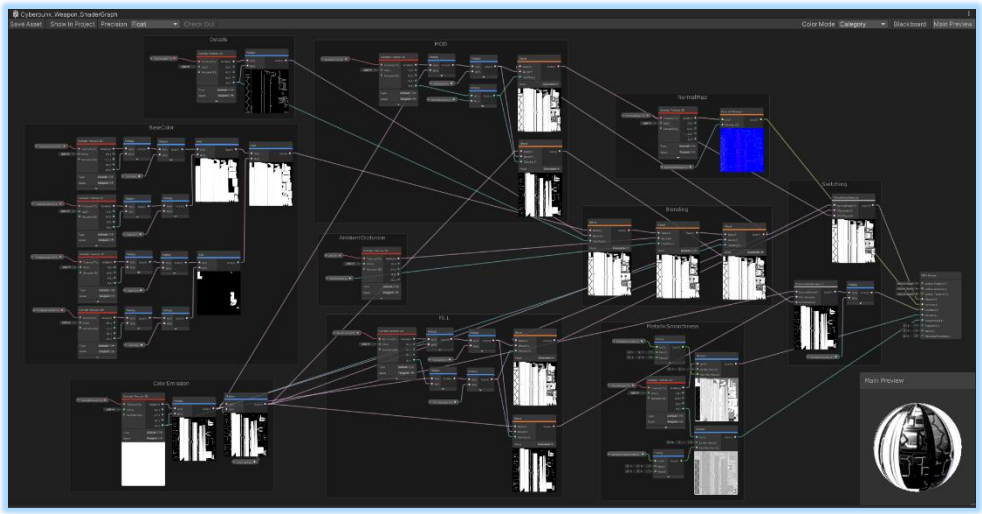
- Go to **Edit** > **Render Pipeline** > **Universal Render Pipeline** > **Upgrade Selected Materials to UniversalRP Materials**





CYBERPUNK_WEAPON_SHADERGRAPH

A shader graph for all weapons. On Unity 2019 and 2020, URP support only. On Unity 2021 Built-In and URP support.

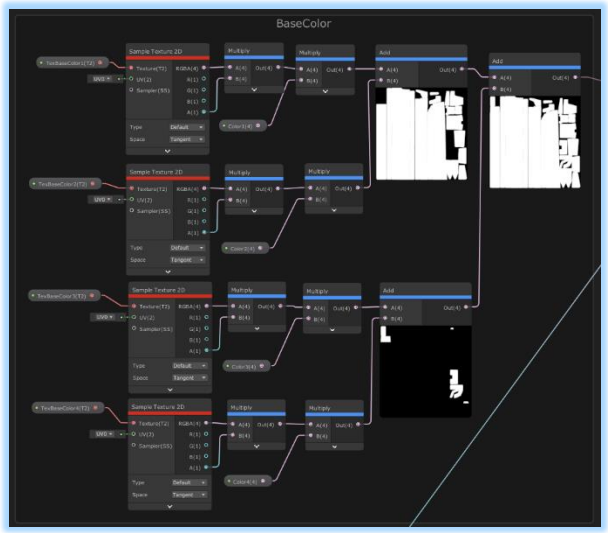


BASECOLOR SECTION

Includes all the fragments of the base texture. Each texture is multiplied by its own **Alpha channel**, then multiplied again to change the color. After color tint, al textures are mixed by **Add nodes**.

Al textures are colorless and FFFFFF color values.

Output: -> Blending section



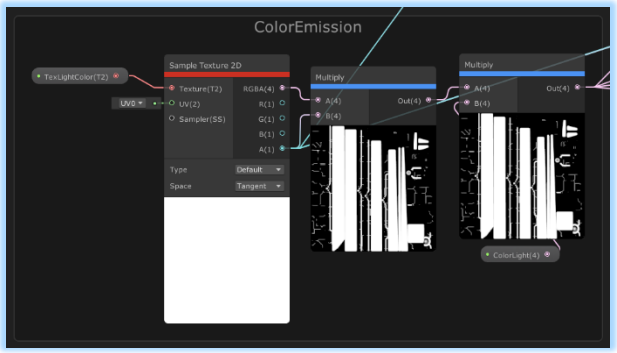
COLOREMISSION SECTION

Section for input the texture that will be used for illuminated parts of the weapons.

It's multiplied by its own **Alpha Channel**, and then multiplied again by a **Color variable** for color change.

Outputs: Alpha Channel -> MOD section
FILL section
Blending section

Multiply node -> MOD section
FILL section
Blending section
Switching section

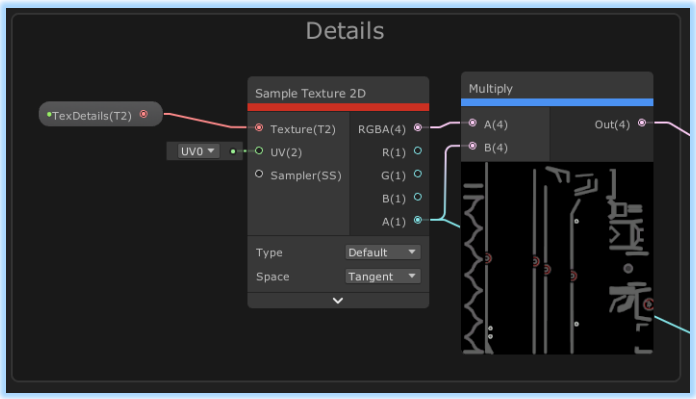


DETAILS SECTION

Section for **Detail texture input**. It's multiplied by its own **Alpha Channel** only. No color changes performed.

Outputs: Alpha Channel -> Blending section

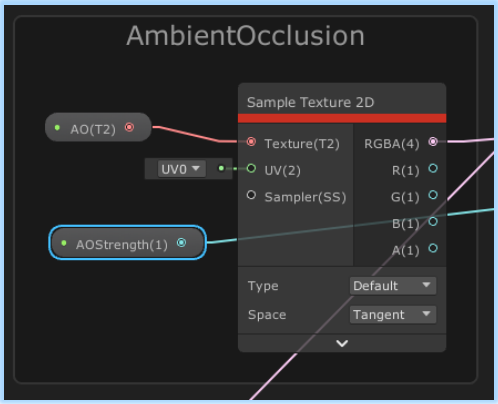
Multiply node -> Blending section



AMBIENTOCCLUSION SECTION

Section for **Ambient Occlusion map input** and **AOStrength variable.**

Outputs: RGBA Channel/AOStrength variable -> Blending section



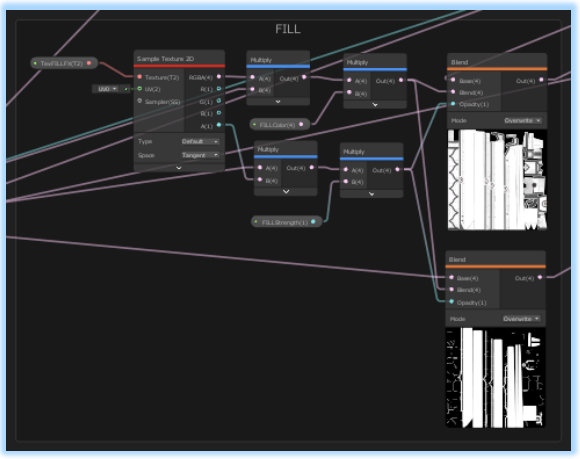
MOD/FILL SECTIONS

Take the textures for both color light effects.

Each one multiplied its texture by the **Alpha Channel** from **ColorEmission section**. Then it's multiplied again by a **Color variable**. Its **Alpha Cannel** is multiplied by a **Float variable** to control the strength of the effect.

Two **Blend nodes** are used for blending both **RGBA Channel (from Blending section)** and **Alpha Channel (from ColorEmission section),** using **Overwrite algorithm,** and its **Alpha Channel** as **Opacity value**.

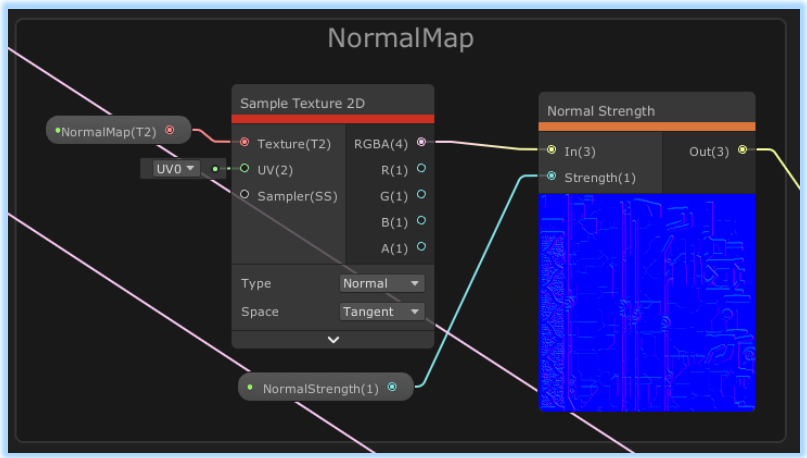
Outputs: -> Switching section



NORMALMAP SECTION

Input section for **Normal Map**. Uses a **Normal Strength node** to control the strength applied of the **Normal Map** on the surface of the material.

Output: -> Normal Material Output



BLENDING SECTION

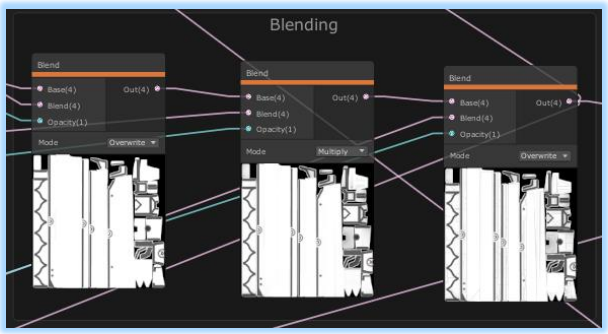
Blends the output from **BaseColor and Details (first step), AmbientOcclusion (second step), and ColorEmission (third step) sections.**

On first step, uses **Overwrite algorithm** and **Alpha Channel** from **Details section** as **Opacity**.

On second step, uses **Multiply algorithm** and **AOStrength variable** from **AmbientOcclusion section** as **Opacity**.

On first step, uses **Overwrite algorithm** and **Alpha Channel** from **ColorEmission section** as **Opacity**.

Outputs: Alpha Channel -> MOD section
FILL section
Switching section



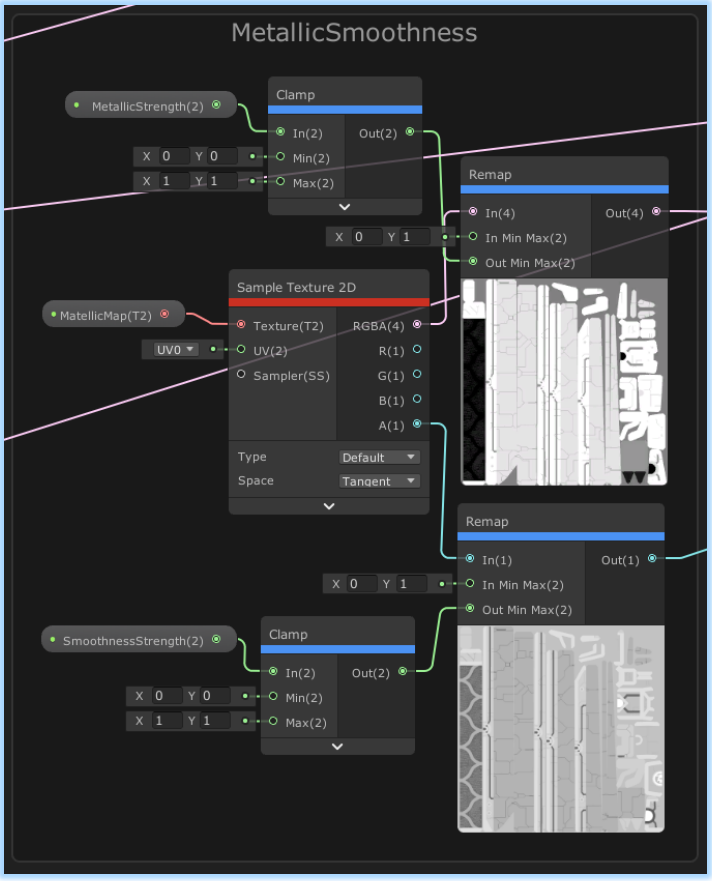
METALLICSMOOTHNESS SECTION

Input for **Metallic map**. Uses **RGBA channel** as **Metallic values** and **Alpha channel** as **Smoothness values.**

Both **Metallic** and **Smoothness** are processed with **Remap nodes** to control the strength. With this configuration you will be able to control the strength in both ways, such as lower values make less metallic and higher values make more metallic on the entire map. The same for **Smoothness values**.

Outputs: -> Metallic material output

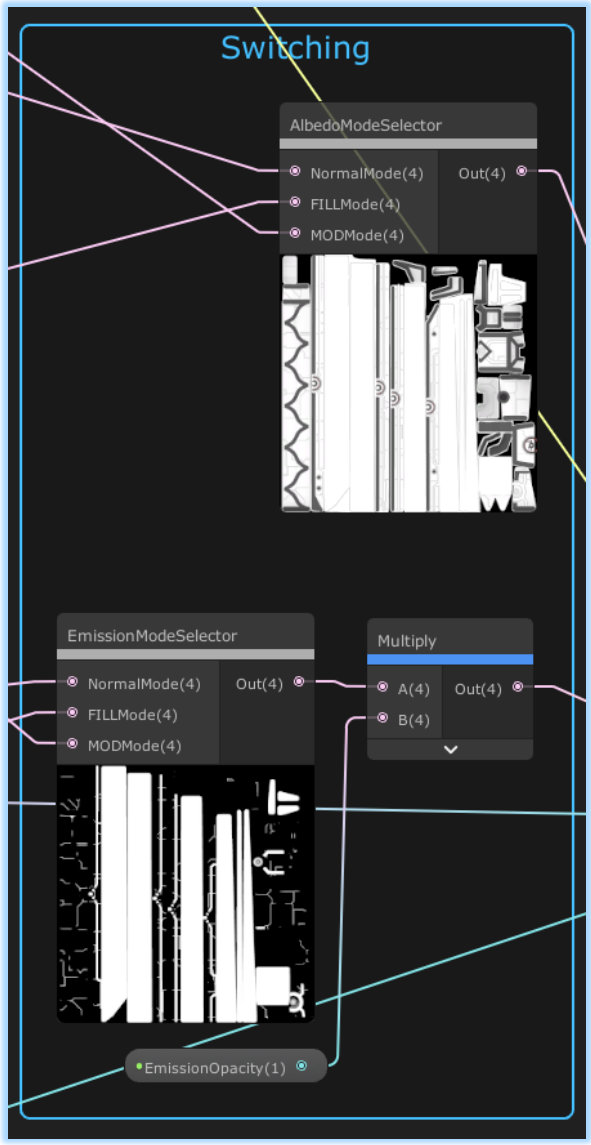
Smoothness material output



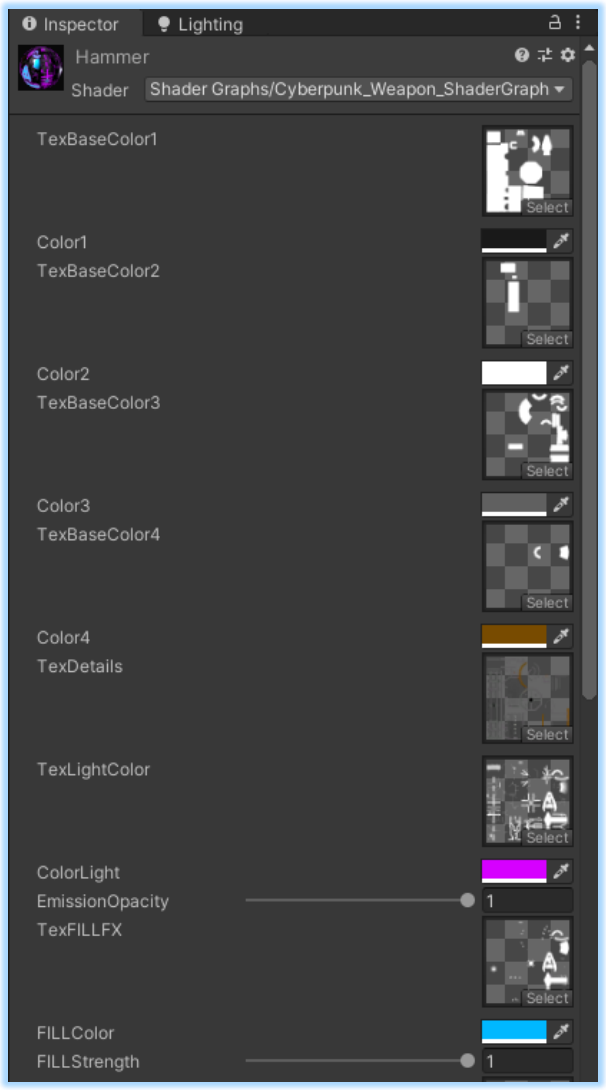
SWITCHING SECTION

Consists of two enumerators, with the purpose of switch between **Normal Mode, FILL Mode and MOD Mode**. One for switching the albedo output and the second for switching the emission output.

Outputs: -> Albedo material output
Emission material output



ON INSPECTOR



- **TexBaseColor 1, 2, 3, 4**
  Texture component of the color base. Up to 4 textures. This **Shader Graph** uses **Add nodes**, so empty slots can be left, but overlapping content between textures must be avoided.

- **Color 1, 2, 3, 4**
  Changes the color of the base texture component. Uses **Multiply nodes** in direct configuration to tint textures, so textures must be #FFFFFF to get all colors.

- **TexDetails**
  A texture slot dedicated for details of the weapons.

- **TexLightColor**
  A texture slot for **Emission maps**. Texture must be #FFFFFF to ensure proper color functionality and must be transparent for proper blending.

- **ColorLight**
  Changes the color of the **Emission map**. Uses **Multiply nodes** in direct configuration to tint it. HDR supported.

- **EmissionOpacity**
  Controls the intensity of the **Emission map**.

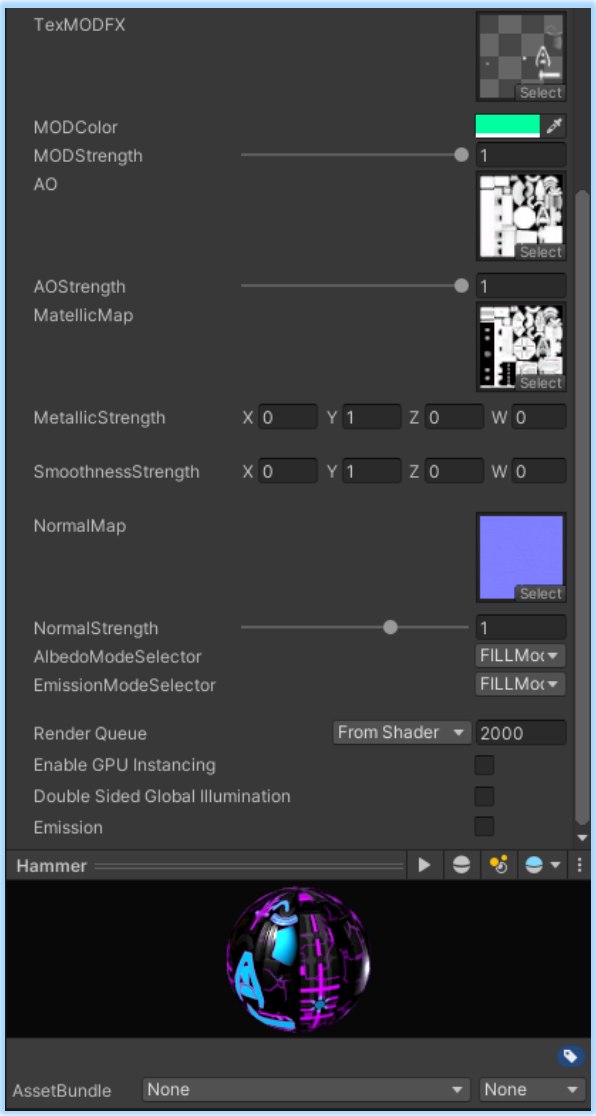
- **TexFILLFX**
  Special **Emission map** that adds the effect **FILL** over emitting parts of the weapon.

- **FILLColor**
  Changes the color of the **FILL effect Emission map**. Uses **Multiply nodes** in direct configuration to tint it.
  HDR supported.

- **FILLStrength**
  Controls the opacity with which the **FILL effect Emission map** is blended over main **Emission map**.

- **TexMODFX**
  Special **Emission map** that adds the effect **MOD** over emitting parts of the weapon.
- **MODColor**
  Changes the color of the **MOD effect Emission map**. Uses **Multiply nodes** in direct configuration to tint it. HDR supported.

- **MODStrength**
  Controls the opacity with which the **MOD effect Emission map** is blended over main **Emission map.**



- **AO**
  Texture slot dedicated for **Ambient Occlusion maps**.

- **AOStrength**
  Controls the opacity of the **Ambient Occlusion map.**

- **MetallicMap**
  Texture slot dedicated for **Metallic maps**. Uses the **standard metallic workflow**, so **grayscale** is used for **Metallic values** and **alpha channel** is used for **Smoothness values**.

- **MetallicStrength**

Controls the strength of the **Metallic values.** It can control both minimum and maximum side of the range.
X controls minimum values.
Y controls maximum values.
*****You can invert the values of the maps by swapping the values between **X and Y.**

- **SmoothnessStrength**
  Controls the strength of the **Smoothness values.** It can control both minimum and maximum side of the range.
  X controls minimum values.
  Y controls maximum values.
  *****You can invert the values of the maps by swapping the values between **X and Y.**

- **NormalMap**
  Texture slot for **Normal maps.**

- **NormalStrength**
  Controls the strength of the **Normal map.**
- **AlbedoModeSelector**
  Enumerator that controls the selection of the effect mode between **Normal mode, FILL mode and MOD mode.** This enumerator is for the **Albedo.**

- **EmissionModeSelector**
  Enumerator that controls the selection of the effect mode between **Normal mode, FILL mode and MOD mode.** This enumerator is for the **Emission.**