# わかる！
# UE4でVRMを扱う仕組み

はるべえ　@ruyo_h

# 自己紹介

- はるべえ　@ruyo_h 
- ゲームプログラマ歴 10年ちょっと

# わかる！　VRM4Uの仕組み

- VRMとは
  - VR向け3Dアバターファイルフォーマット


- VRM4Uとは
  - VRMファイルを扱うためのUE4プラグイン
  （Unityでいうところの UniVRM）

UNREAL
ENGINE

# 動画


File Edit Window Help
Modes
Save Current
Source Control
Content
Marketplace
Settings
Blueprints
Cinematics
Build
Compile
Pause
Stop
Possess
Viewport 1
Perspective
Lit
Show
LIGHTING NEEDS TO BE REBUILT (663 unbuilt objects)
Recently Placed
Basic
Lights
Cinematic
Visual Effects
Geometry
Volumes
All Classes
Empty Actor
Empty Character
Empty Pawn
Point Light
Player Start
Cube
Sphere
Cylinder
Cone
Plane
Box Trigger
Sphere Trigger
World Outliner
973 actors (1 selected)
Details
World Settings
Select an object to view details.
Content Browser
Output Log
Add New
Import
Save All
Content
_pp1
NewFolder4
Texture
Node Mapping Container
Vrm License Object
Vrm Meta Object
Vrm Asset List Object
69 items (1 selected)
View Options

おわかりいただけただろうか

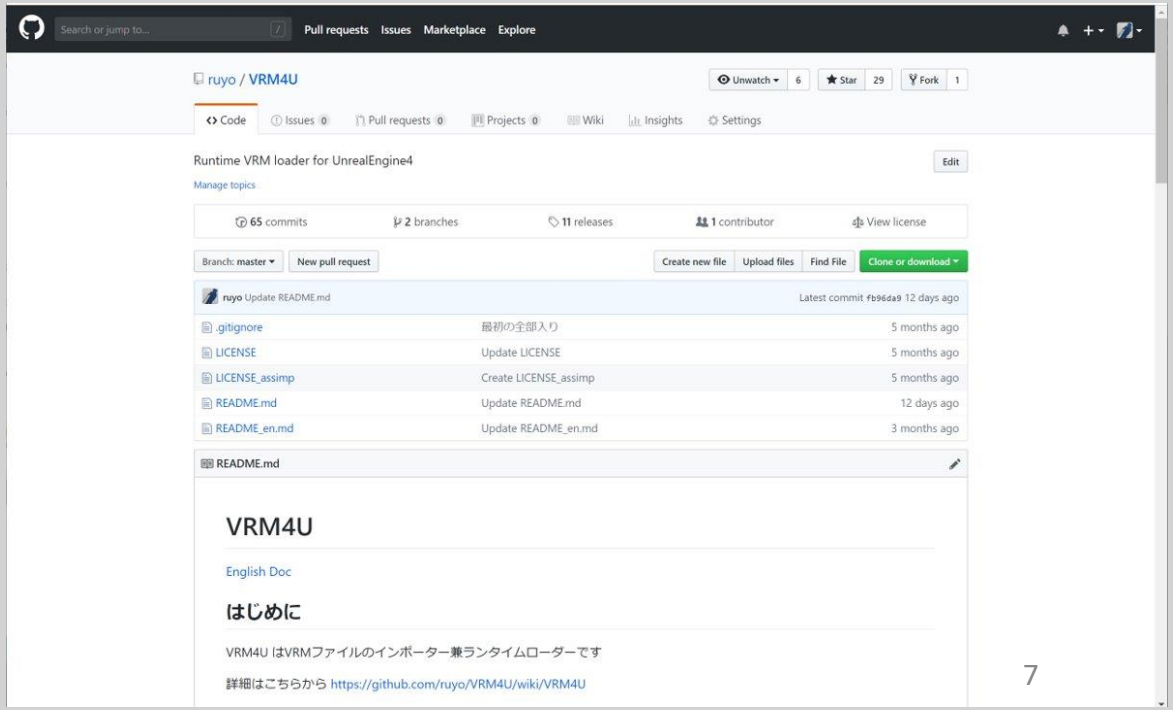
# おわかりいただけただろうか

- VRMのインポート
- PBR背景＋NPRキャラクタ
- ランタイムリターゲット
- 揺れ骨（髪）
- BlendShape（顔アニメ）
- NPRパラメータ調整

# VRM4U、Githubで公開中

- https://github.com/ruyo/VRM4U
- 導入はPlugins/VRM4U に配置すれば完了

- 動作環境
  - Windows、UE4.19～4.22
  - Android（要ビルド）



# アジェンダ

- VRM4Uの方針
- マテリアル
- インポータ
- アニメーション
- モバイル対応
- まとめ

# アジェンダ

- VRM4Uの方針
- マテリアル
- インポータ
- アニメーション
- モバイル対応
- まとめ

# VRM4Uの方針

# 個人的な こだわりポイント

- UE4でVRMを
  - 手軽に利用したい
  - ゲームで使いたい
  - モバイルで使いたい
- キレイなトゥーンを出したい
- UE4の良さに乗っかりたい

# 個人的な こだわりポイント

- UE4でVRMを
  - 手軽に利用したい
  - ゲームで使いたい
  - モバイルで使いたい
- キレイなトゥーンを出したい
- UE4の良さに乗っかりたい

プラグインで提供する

エフェクト・半透明と相性が良い
既存のアセットを利用しやすい

Forward/Deferredどちらでも動作する

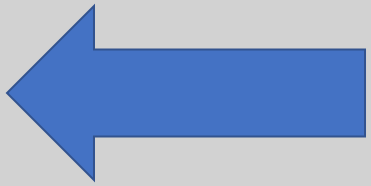
UE4の機能を積極的に利用する
既存機能と競合しないようにする
~~打倒Unity~~

# 実装の方針

- プラグインで完結する。
  - Engineは改造しない
- マテリアルで完結する。
  - Diferred/Forwardどちらでも描画できる。
  - ポストエフェクトを利用しない。
- UE4の機能・利点を活かす。
  - PBR
  - HumanoidRig
  - PhysicsAsset

# アジェンダ

- VRM4Uの方針
- マテリアル
- インポータ
- アニメーション
- モバイル対応
- まとめ



# マテリアル

# マテリアル　目標

- MToon（VRM標準シェーダ）を再現できればOK！

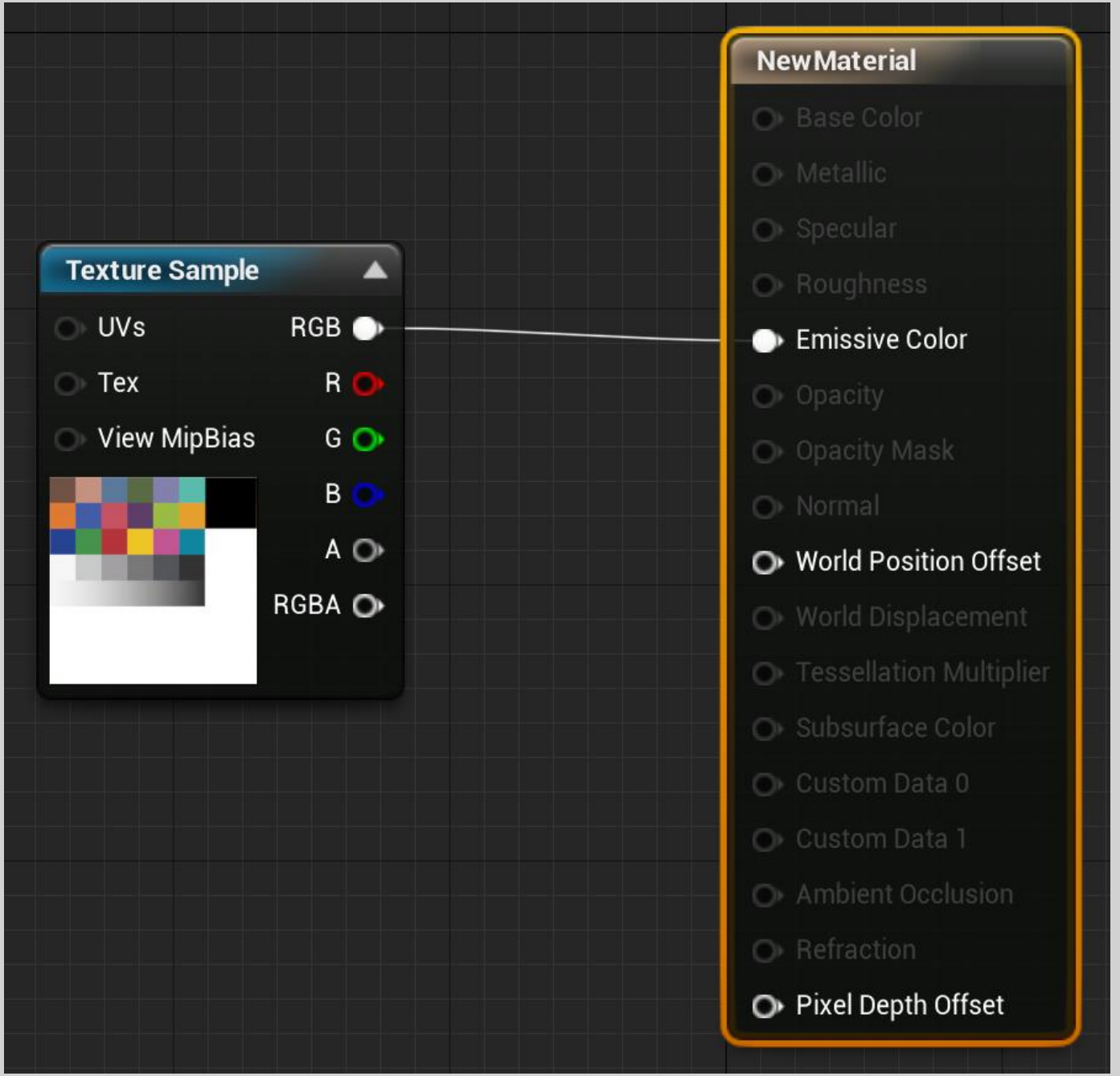
- 制約
  - モバイルでも動作する（Diferred/Forward両方対応）
  - PBRと両立させる（ポストプロセス、GBufferを利用しない）

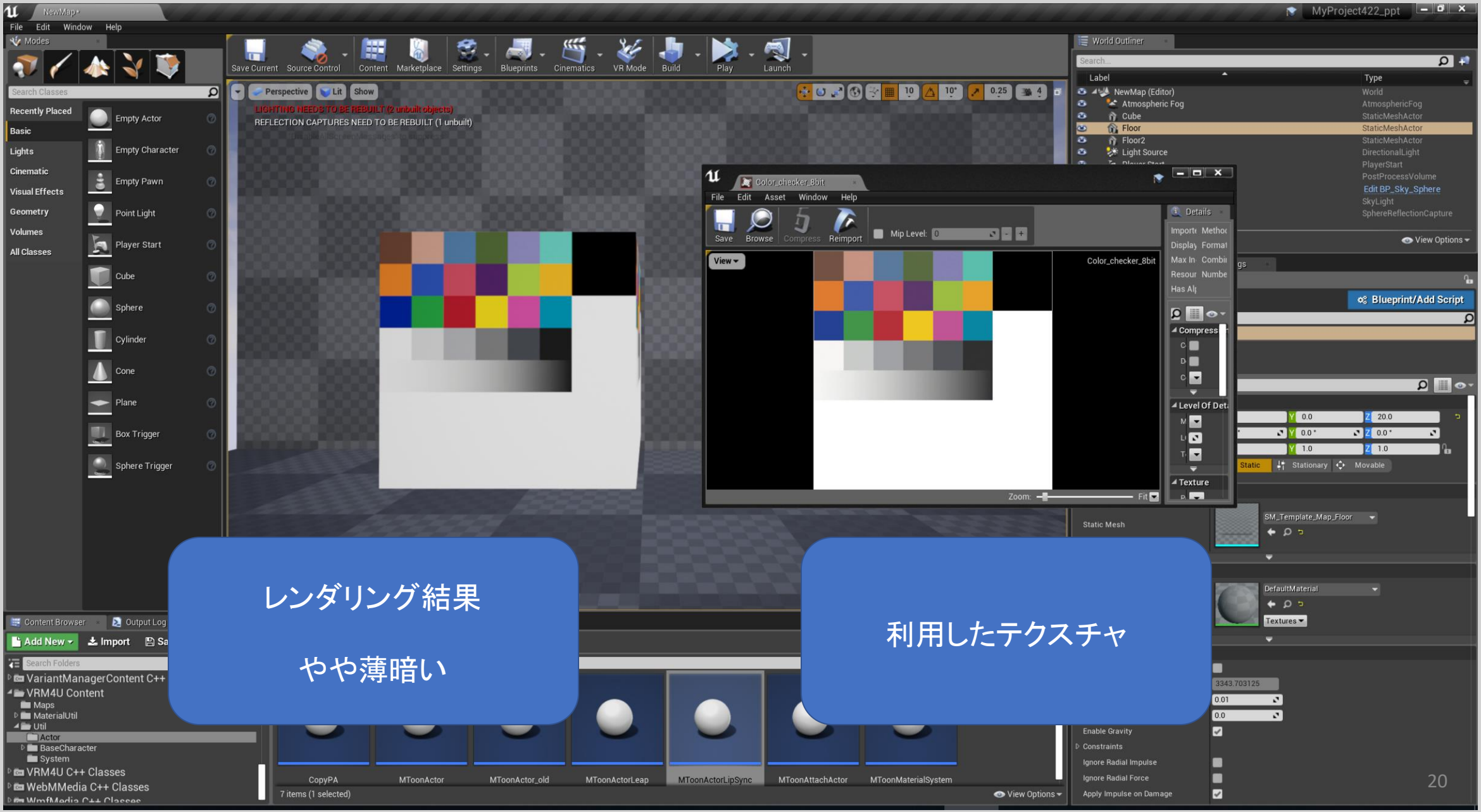
# MToonの主な機能

- 主/陰 を任意の色・テクスチャで描画できる。
- 陰影の境界値を調整できる。
- 輪郭線
  - 色を変更できる。マテリアル毎。
  - 太さを変更できる。マテルアル毎。テクスチャで部位単位に細かく調整。
- 他にもいろいろ・・・

# UE4におけるマテリアルの課題

意図どおりの色（数値）を出力することが難しい


Texture Sample
UVs
Tex
View MipBias
RGB
R
G
B
A
RGBA
NewMaterial
Base Color
Metallic
Specular
Roughness
Emissive Color
Opacity
Opacity Mask
Normal
World Position Offset
World Displacement
Tessellation Multiplier
Subsurface Color
Custom Data 0
Custom Data 1
Ambient Occlusion
Refraction
Pixel Depth Offset


レンダリング結果
やや薄暗い
利用したテクスチャ


RGB=(219,197,23)
RGB=(238,199,33)
レンダリング結果
やや薄暗い
利用したテクスチャ

# 意図どおりの色を出すには

- トーンマップを無効化する
- ポストプロセスでBaseColorを利用する
- 古いトーンマップに切り替える
  - ちょっと色ズレします

# 意図どおりの色を出すには

- トーンマップを無効化する

いくつかポストプロセスが無効化される
（ColorGrading,Bloom,LensFlareなど、、）

- ポストプロセスでBaseColorを利用する

半透明を解決できない
TemporalAAだとブレる

- 古いトーンマップに切り替える
  - ちょっと色ズレします

機能的に大きな問題なし
FilmicToneMapperが使えないのは残念

# 意図どおりの色を出すには

- トーンマップを無効化する

いくつかポストプロセスが無効化される
（ColorGrading,Bloom,LensFlareなど、、）

- ポストプロセスでBaseColorを利用する

半透明を解決できない
TemporalAAだとブレる

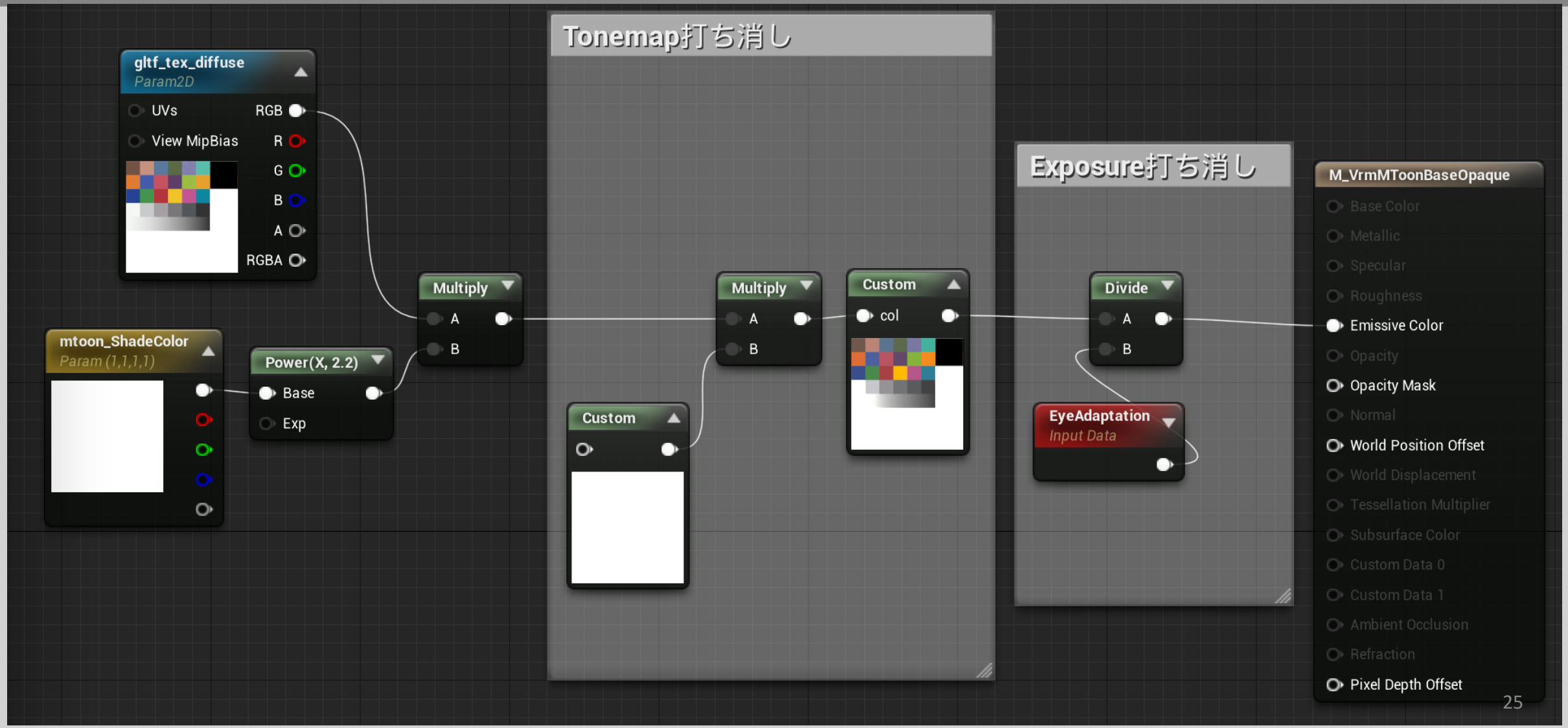
- 古いトーンマップに切り替える
  - ちょっと色ズレします

機能的に大きな問題なし
FilmicToneMapperが使えないのは残念

- トーンマップを逆変換する（VRM4U）
  - ちょっと色ズレします

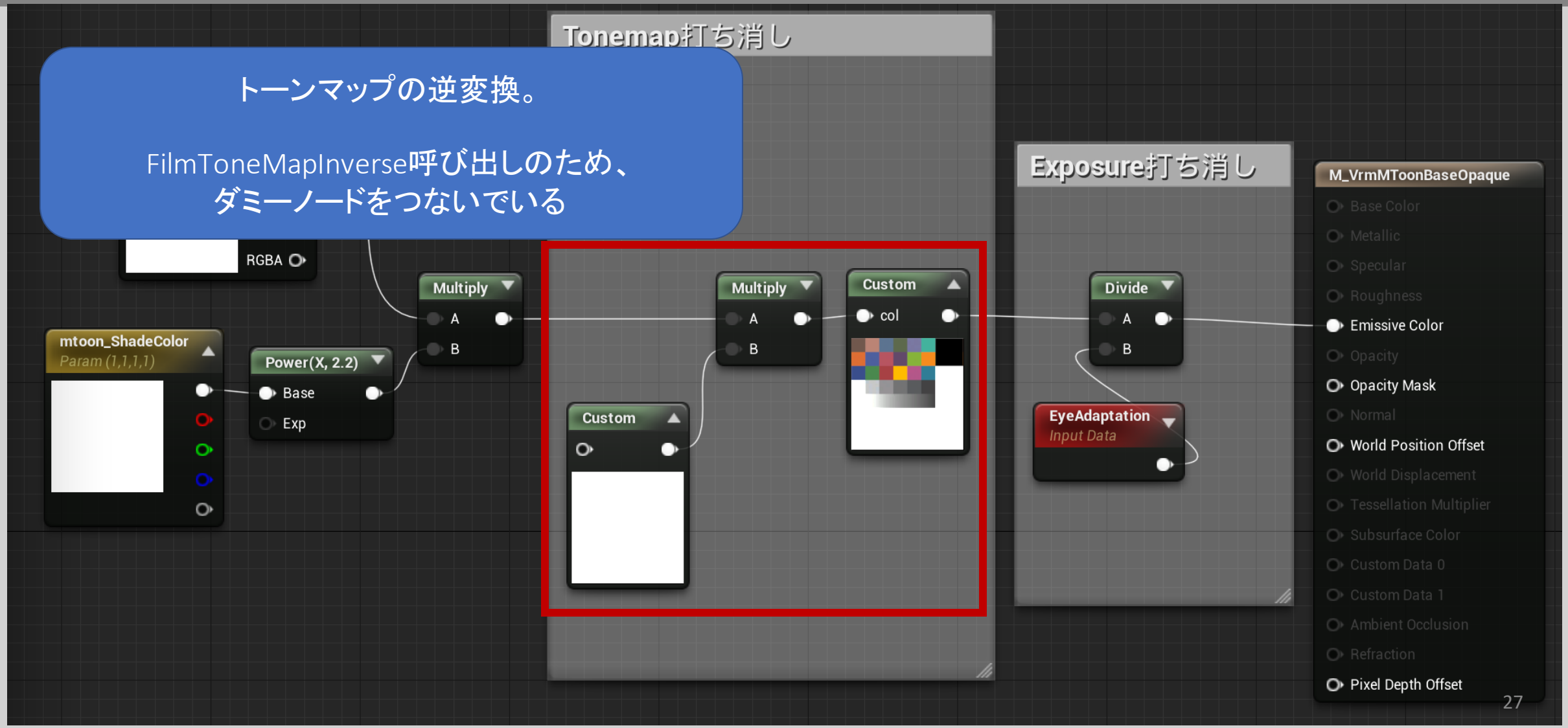
NEW

# VRM4Uの基本ノード

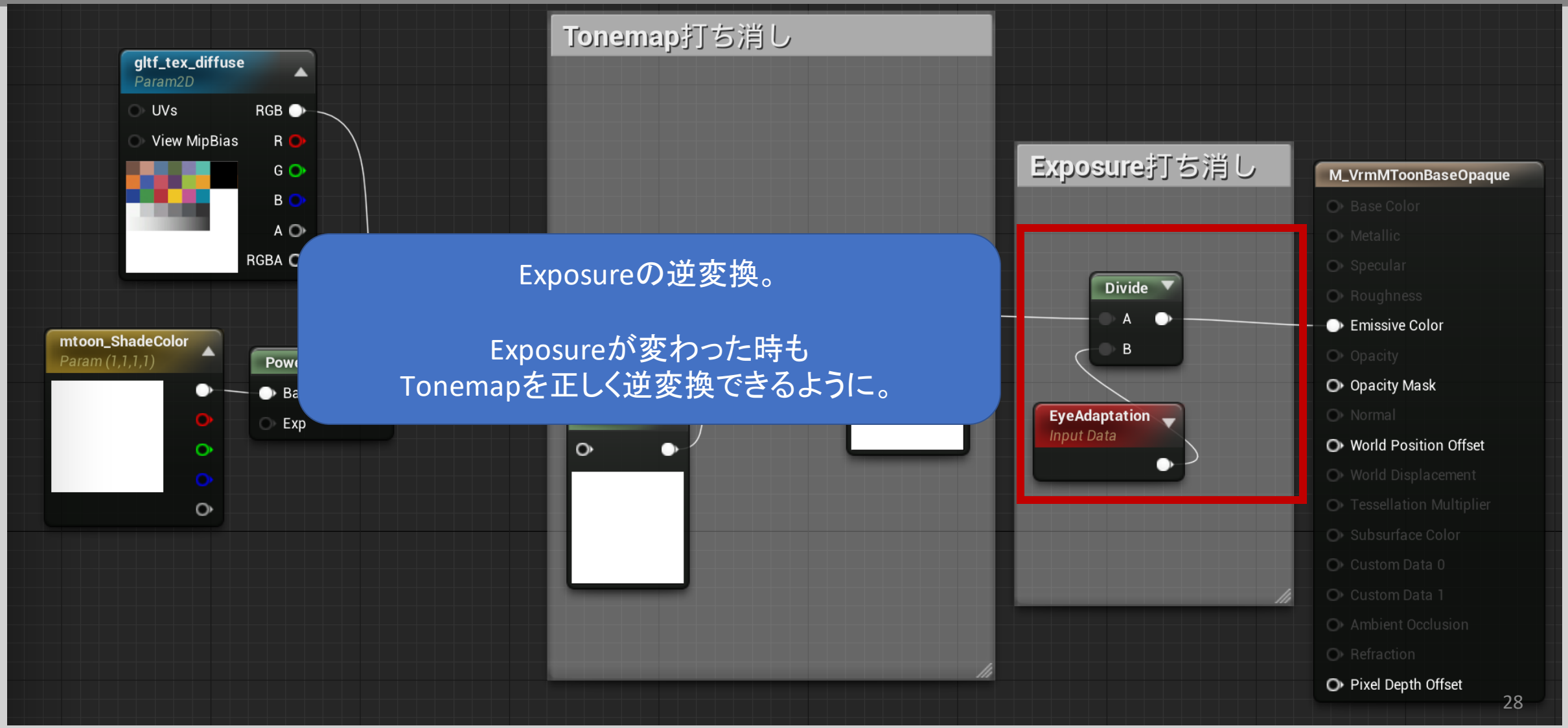


# VRM4Uの基本ノード

# VRM4Uの基本ノード



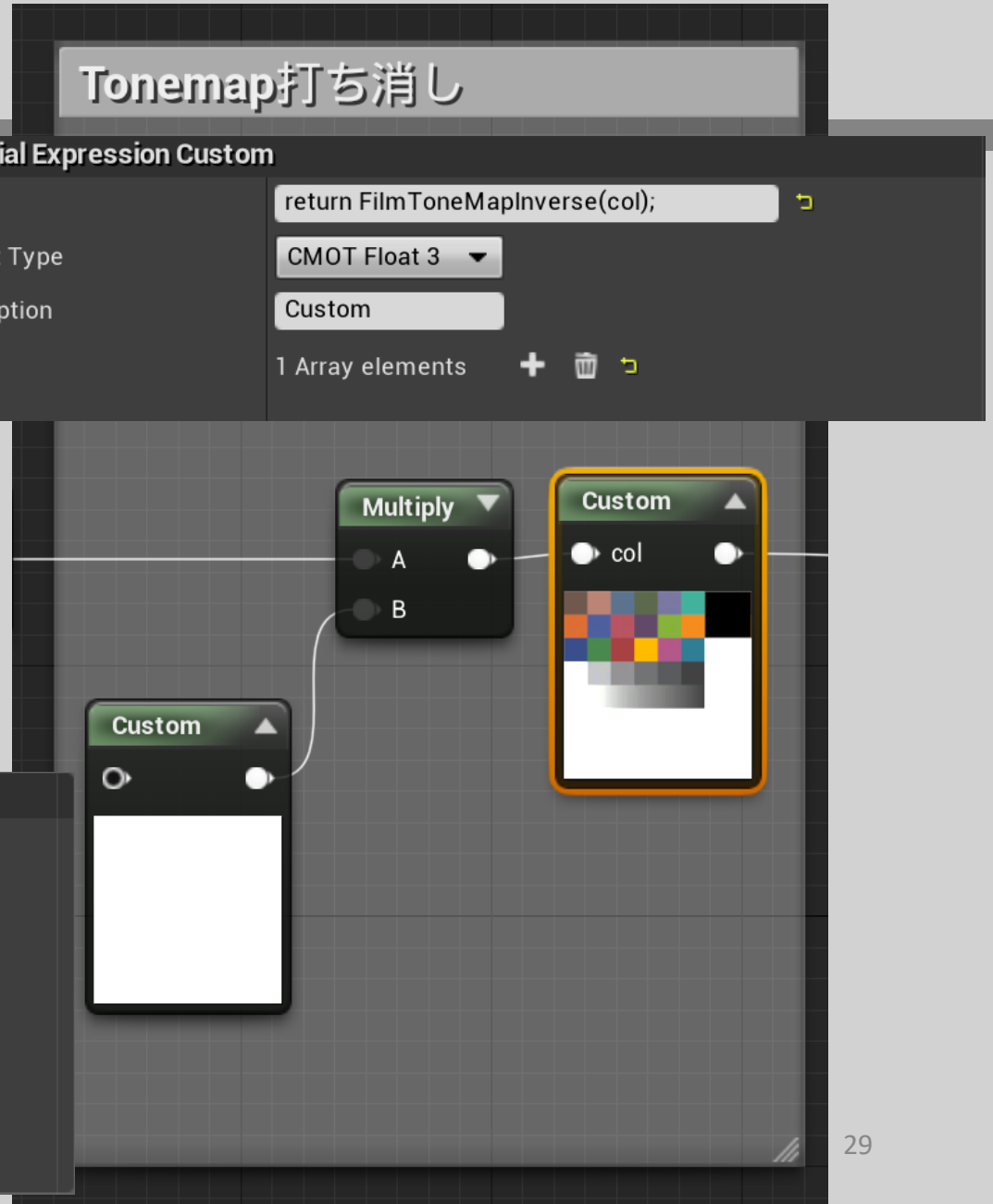
# VRM4Uの基本ノード

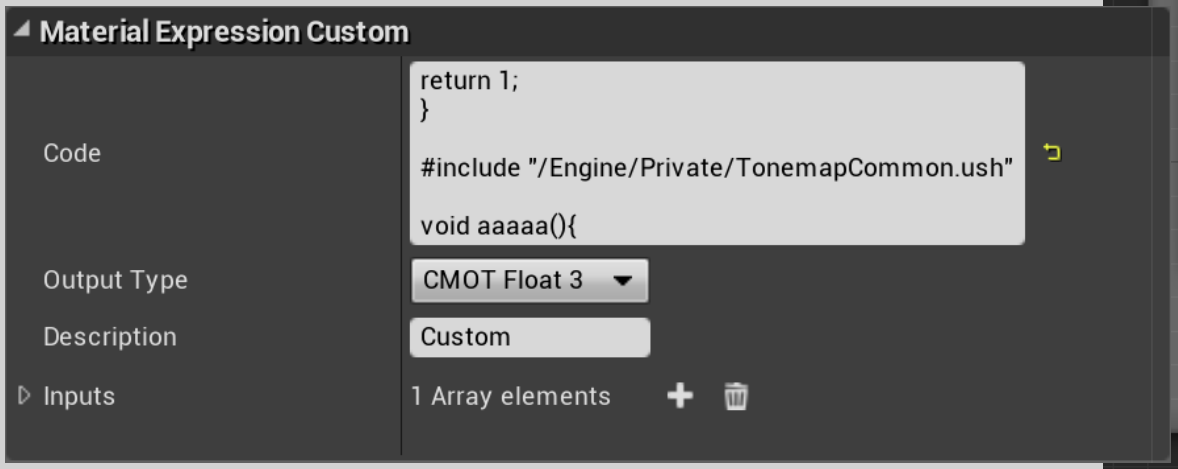


# FilmToneMapInverse

- Tonemapを逆変換してくれる
  - パラメータは固定。テスト実装？
    - Engine内のどこからも呼び出されていない

（詳しい方、正確な逆変換関数の
実装をお願いします…）




Perspective
Lit
Show
LIGHTING NEEDS TO BE REBUILT (3 unbuilt objects)
REFLECTION CAPTURES NEED TO BE REBUILT (1 unbuilt)
'DisableAllScreenMessages' to suppress
10
10°
0.25
4
補正なし
やや薄暗い
Tonemap, Exposure補正あり
Selected Actor(s) in: NewMap (Persistent)
Level: NewMap (Persistent)

# ライトの反映

- SkyLight、DirectionalLightならば手軽に反映できそう

- SkyLight
  - usf内をGetSkySHDiffuseSimple, GetSkySHDiffuse で検索

- DirectionalLight
  - ResolvedView.DirectionalLightColor　など。SceneView.h を参照。
  - 光源と法線より、主/陰 を塗り分けする

- SkyLightは法線方向に対応した色が返る。
  - 陰影がついてしまうので注意。

# MatCap

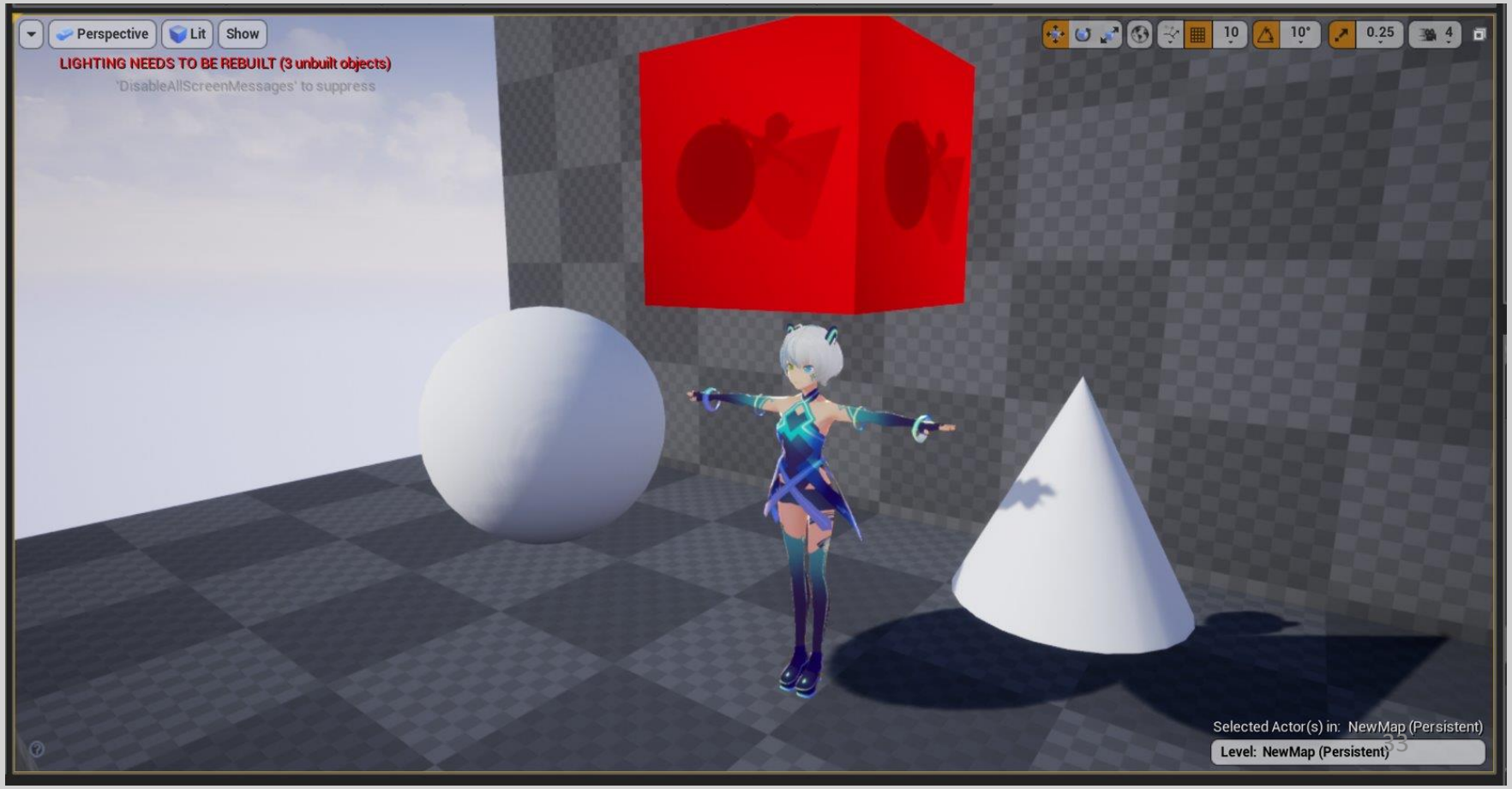
- 法線とカメラベクトルによる効果
- MToonではリムライトの効果として利用されることが多い


MatCapなし


MatCapあり

# 影の反映

- 影領域を判別するため、シャドウマップを生成しています




よく観察すると
影が粗い…

# 輪郭線を描画するには

- ポストプロセスで
  （カスタム）デプスを利用

- マテリアルで頑張る（VRM4U）

輪郭線の色・太さを個別に制御するのが難しく、
MToonの機能を再現しきれない

# 輪郭線の描画

- PoseableMeshを利用した背面法
  - ヒストリアさんの解説へ
    - http://historia.co.jp/archives/5587/

- VRM4Uで追加
  - 輪郭線を個別に制御（個別のOutline用マテリアルを生成）
  - 画角、解像度による影響の打ち消し
  - カリング方向の逆転（PoseableMeshのスケールをx-1）
    - 苦肉の負荷軽減…


モデル＋輪郭線


モデルを非表示にして、
輪郭線のみを表示したもの

# マテリアルまとめ

- MToonを再現できるようになった
  - FilmToneMapInverse
    - 色をコントロールできるようになった
  - Shadowmap
    - オブジェクトの影、セルフシャドウを受けられるようになった
  - Outlineの再現もOK

- 実装はマテリアルで完結。
  既存のアセットと組み合わせしやすくなった。


Perspective
Lit
Show
10
10°
0.25
4
Level: Room_run (Persistent)

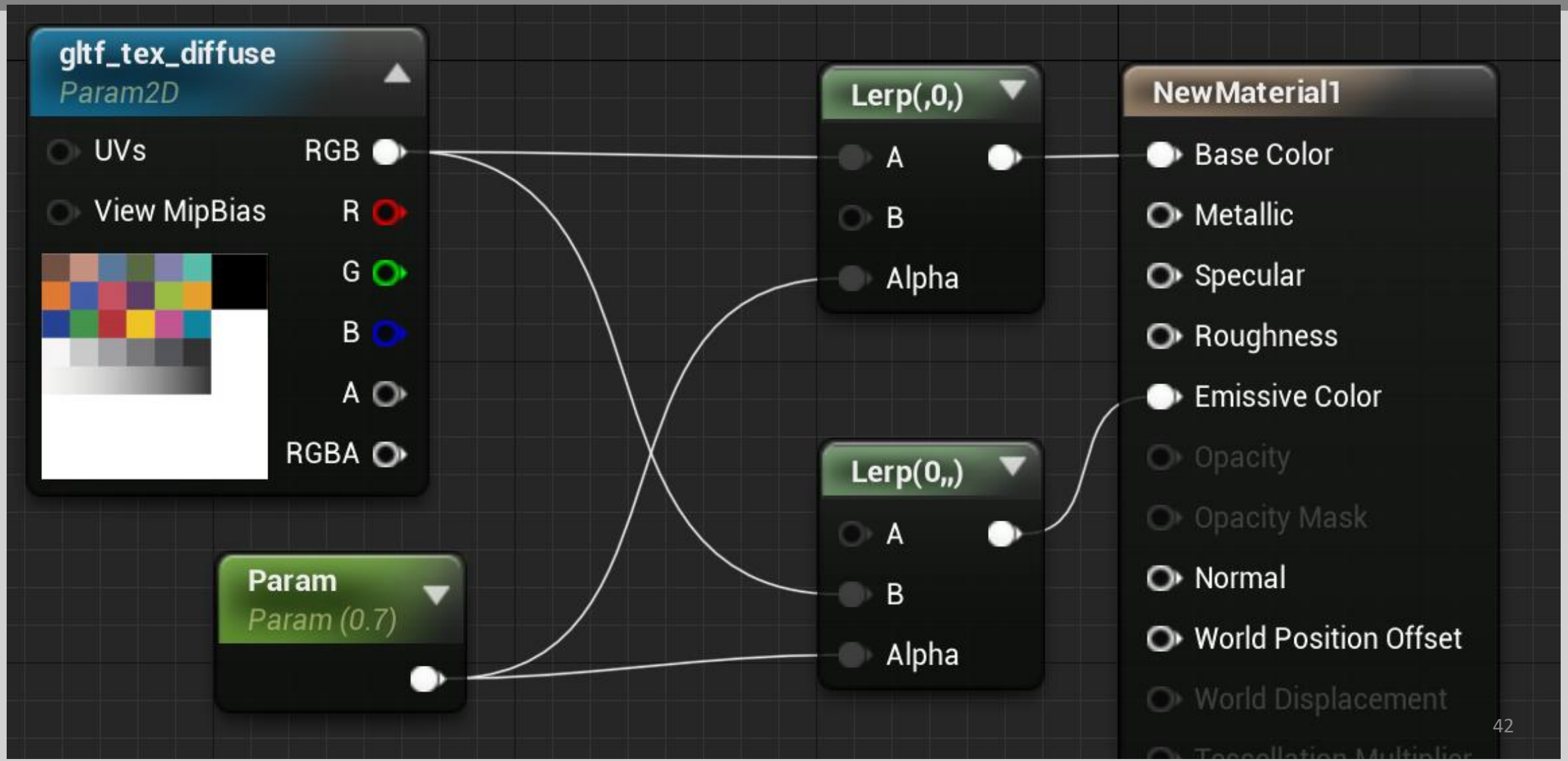
# マテリアル つづきます

# マテリアル 飽くなき欲求

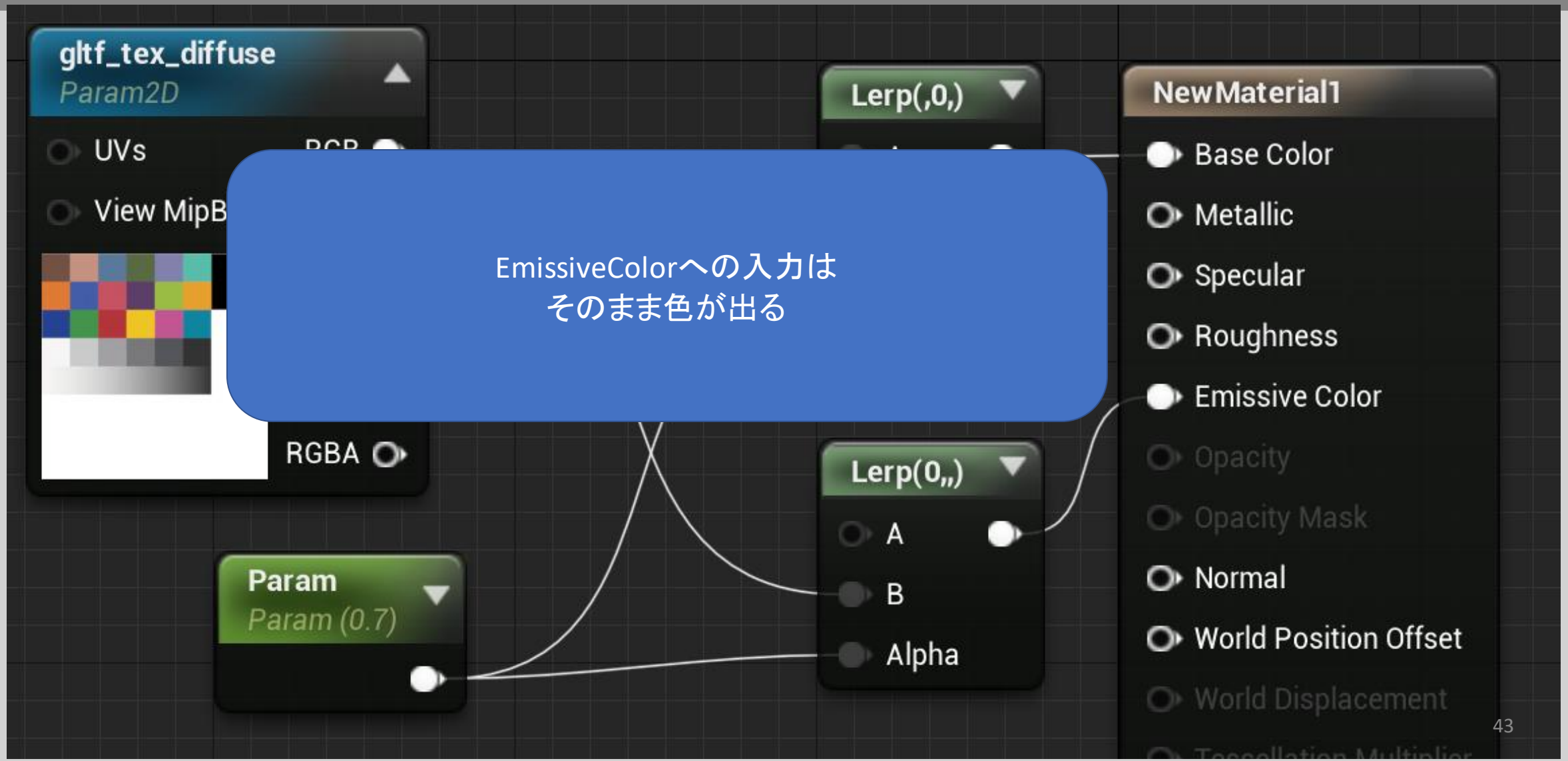
- もっと多くのライトを反映したい
- AOを入れたい
- レイトレーシングと組み合わせたい

Lit版を作りました

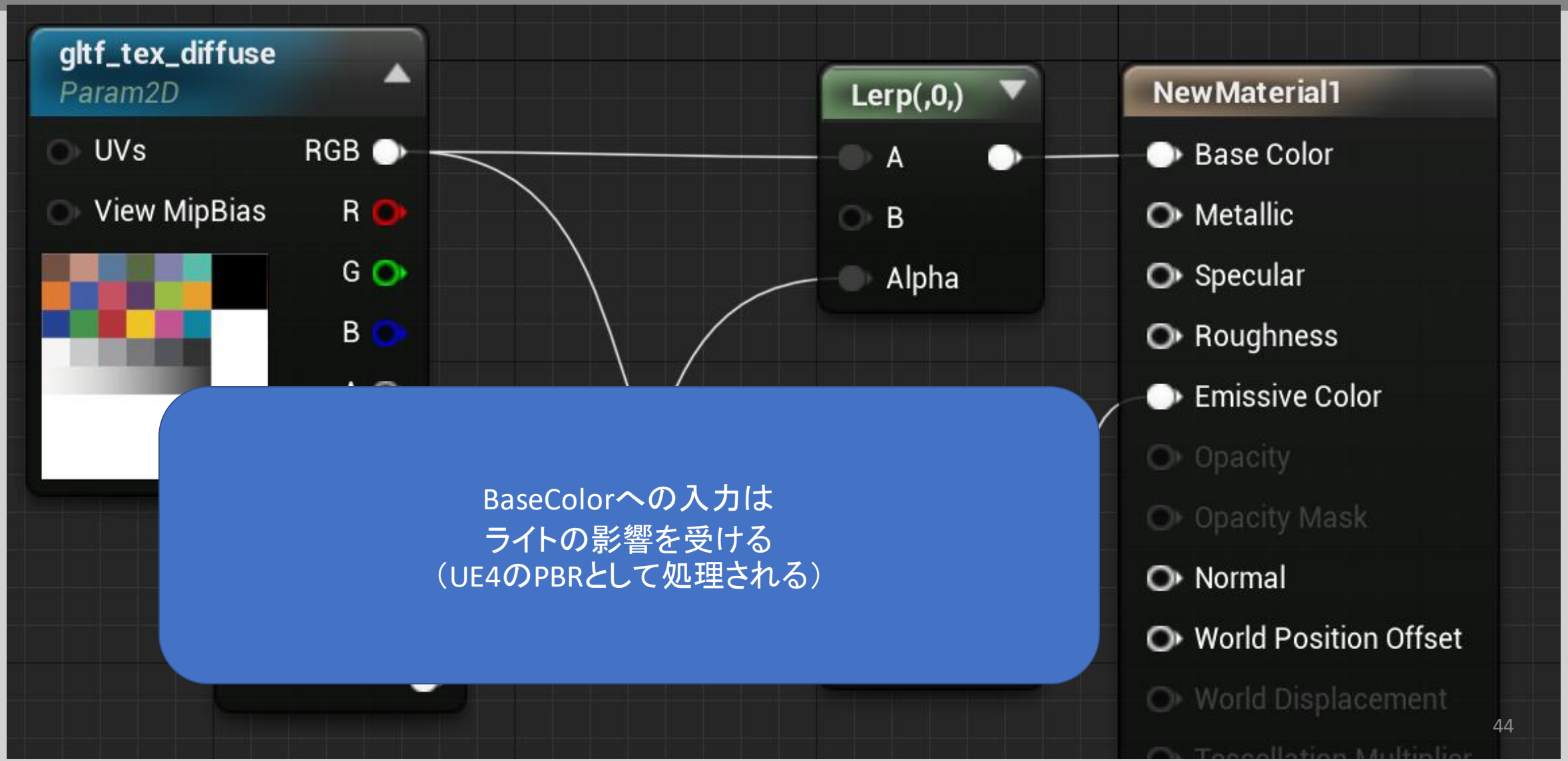
# VRM4UのLIT基本ノード



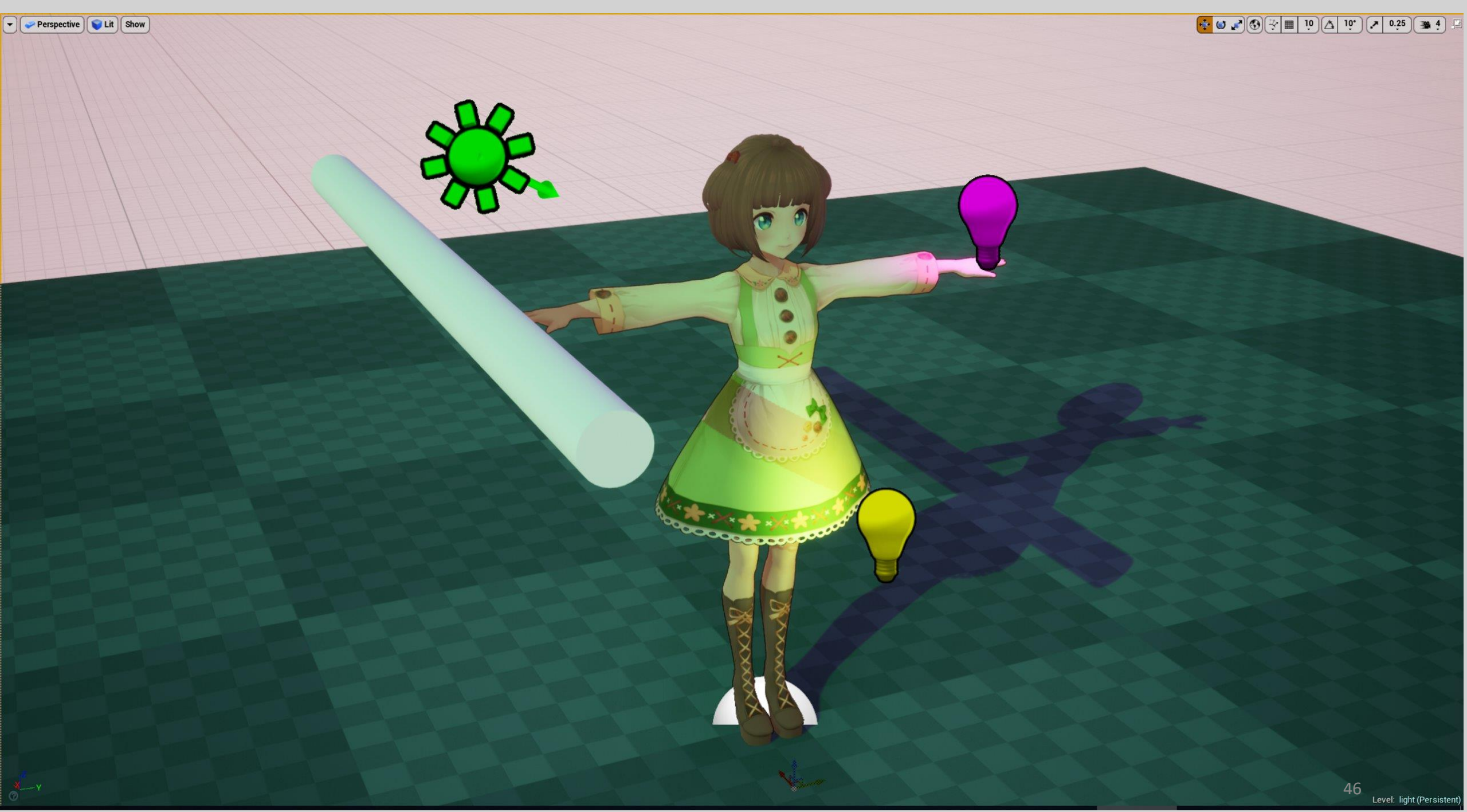
# VRM4UのLIT基本ノード

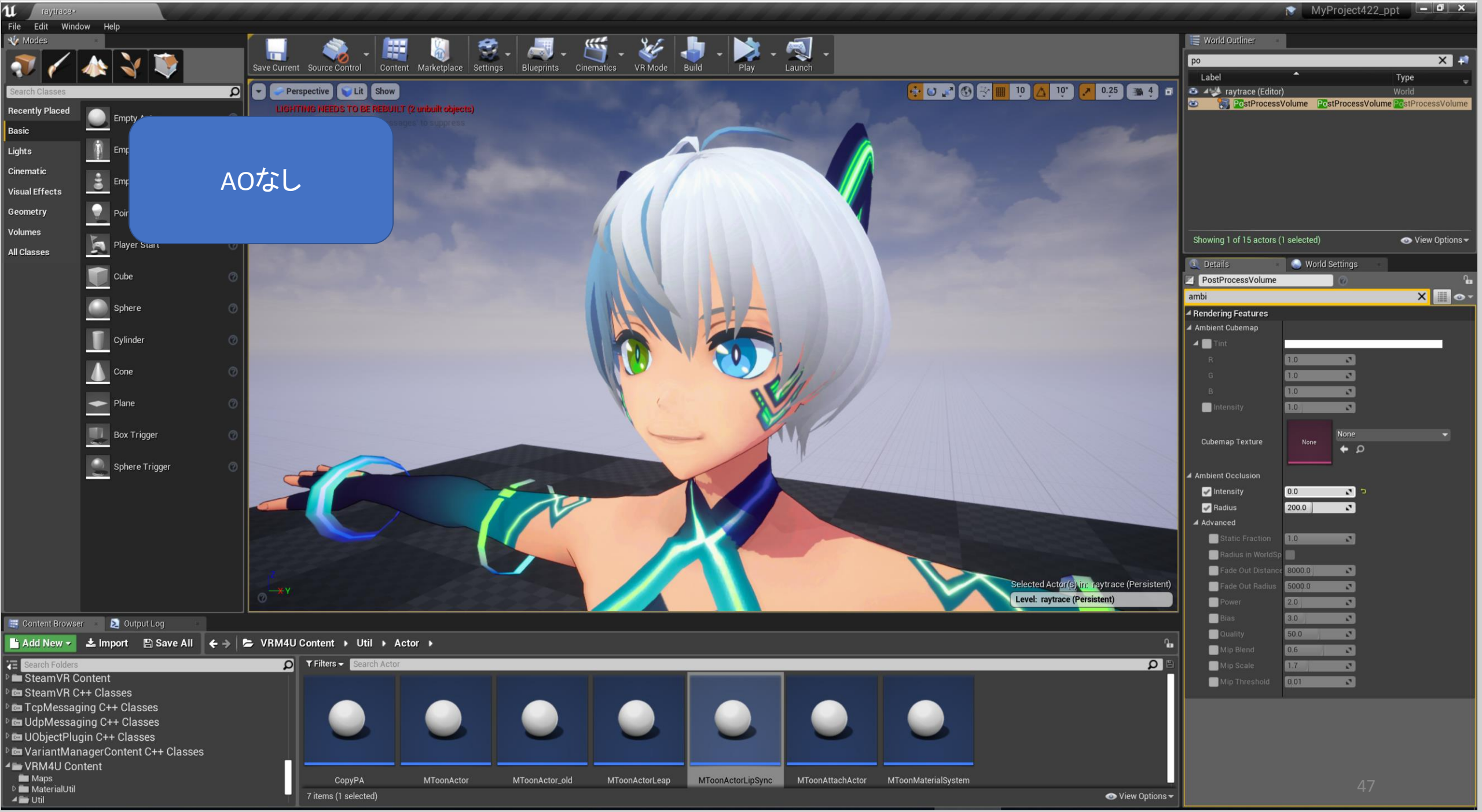


# VRM4UのLIT基本ノード




Perspective
Lit
Show
10
10°
0.25
4
Unlit
Lit
BaseColor x0.2
EmissiveColor x0.8
Lit
BaseColor x1.0
EmissiveColor x0.0
Selected Actor(s) in: Untitled_1 (Persistent)
Level: Untitled_1 (Persistent)


Perspective
Lit
Show
10
10°
0.25
4
46
Level: light (Persistent)


AOなし
raytrace*
MyProject422_ppt
File Edit Window Help
Modes
Save Current
Source Control
Content
Marketplace
Settings
Blueprints
Cinematics
VR Mode
Build
Play
Launch
Perspective
Lit
Show
LIGHTING NEEDS TO BE REBUILT (2 unbuilt objects)
Recently Placed
Basic
Lights
Cinematic
Visual Effects
Geometry
Volumes
All Classes
Player Start
Cube
Sphere
Cylinder
Cone
Plane
Box Trigger
Sphere Trigger
World Outliner
raytrace (Editor)
PostProcessVolume
Showing 1 of 15 actors (1 selected)
View Options
Details
World Settings
PostProcessVolume
ambi
Rendering Features
Ambient Cubemap
Tint
Intensity
Cubemap Texture
Ambient Occlusion
Intensity
Radius
Advanced
Selected Actor(s) in: raytrace (Persistent)
Level: raytrace (Persistent)
Content Browser
Output Log
Add New
Import
Save All
VRM4U Content
Util
Actor
SteamVR Content
SteamVR C++ Classes
TcpMessaging C++ Classes
UdpMessaging C++ Classes
UObjectPlugin C++ Classes
VariantManagerContent C++ Classes
VRM4U Content
Maps
MaterialUtil
Util
CopyPA
MToonActor
MToonActor_old
MToonActorLeap
MToonActorLipSync
MToonAttachActor
MToonMaterialSystem
7 items (1 selected)
View Options
47


raytrace*
MyProject422_ppt
File Edit Window Help
Modes
Save Current
Source Control
Content
Marketplace
Settings
Blueprints
Cinematics
VR Mode
Build
Play
Launch
Perspective
Lit
Show
LIGHTING NEEDS TO BE REBUILT (2 unbuilt objects)
Recently Placed
Basic
Lights
Cinematic
Visual Effects
Geometry
Volumes
All Classes
Player Start
Cube
Sphere
Cylinder
Cone
Plane
Box Trigger
Sphere Trigger
レイトレーシングAO
World Outliner
raytrace (Editor)
PostProcessVolume
Showing 1 of 15 actors (1 selected)
View Options
Details
World Settings
PostProcessVolume
ambi
Rendering Features
Ambient Cubemap
Cubemap Texture
Ambient Occlusion
Intensity
Radius
Advanced
Selected Actor(s) in: raytrace (Persistent)
Level: raytrace (Persistent)
Content Browser
Output Log
Add New
Import
Save All
VRM4U Content
Util
Actor
SteamVR Content
SteamVR C++ Classes
TcpMessaging C++ Classes
UdpMessaging C++ Classes
UObjectPlugin C++ Classes
VariantManagerContent C++ Classes
VRM4U Content
Maps
MaterialUtil
Util
CopyPA
MToonActor
MToonActor_old
MToonActorLeap
MToonActorLipSync
MToonAttachActor
MToonMaterialSystem
7 items (1 selected)
View Options


raytrace*
MyProject422_ppt
File Edit Window Help
Modes
Save Current
Source Control
Content
Marketplace
Settings
Blueprints
Cinematics
VR Mode
Build
Play
Launch
Search Classes
Recently Placed
Basic
Lights
Cinematic
Visual Effects
Geometry
Volumes
All Classes
Empty Actor
Cube
Sphere
Cylinder
Cone
Plane
Box Trigger
Sphere Trigger
SSAO
Perspective
Lit
Show
LIGHTING NEEDS TO BE REBUILT (2 unbuilt objects)
Selected Actor(s) in: raytrace (Persistent)
Level: raytrace (Persistent)
World Outliner
Label
Type
raytrace (Editor)
PostProcessVolume
Showing 1 of 15 actors (1 selected)
View Options
Details
World Settings
PostProcessVolume
ambi
Rendering Features
Ambient Cubemap
Tint
Intensity
Cubemap Texture
None
Ambient Occlusion
Intensity
Radius
Advanced
Static Fraction
Radius in WorldSp
Fade Out Distance
Fade Out Radius
Power
Bias
Quality
Mip Blend
Mip Scale
Mip Threshold
Content Browser
Output Log
Add New
Import
Save All
VRM4U Content
Util
Actor
Search Folders
SteamVR Content
SteamVR C++ Classes
TcpMessaging C++ Classes
UdpMessaging C++ Classes
UObjectPlugin C++ Classes
VariantManagerContent C++ Classes
VRM4U Content
Maps
MaterialUtil
Util
Filters
Search Actor
CopyPA
MToonActor
MToonActor_old
MToonActorLeap
MToonActorLipSync
MToonAttachActor
MToonMaterialSystem
7 items (1 selected)
View Options

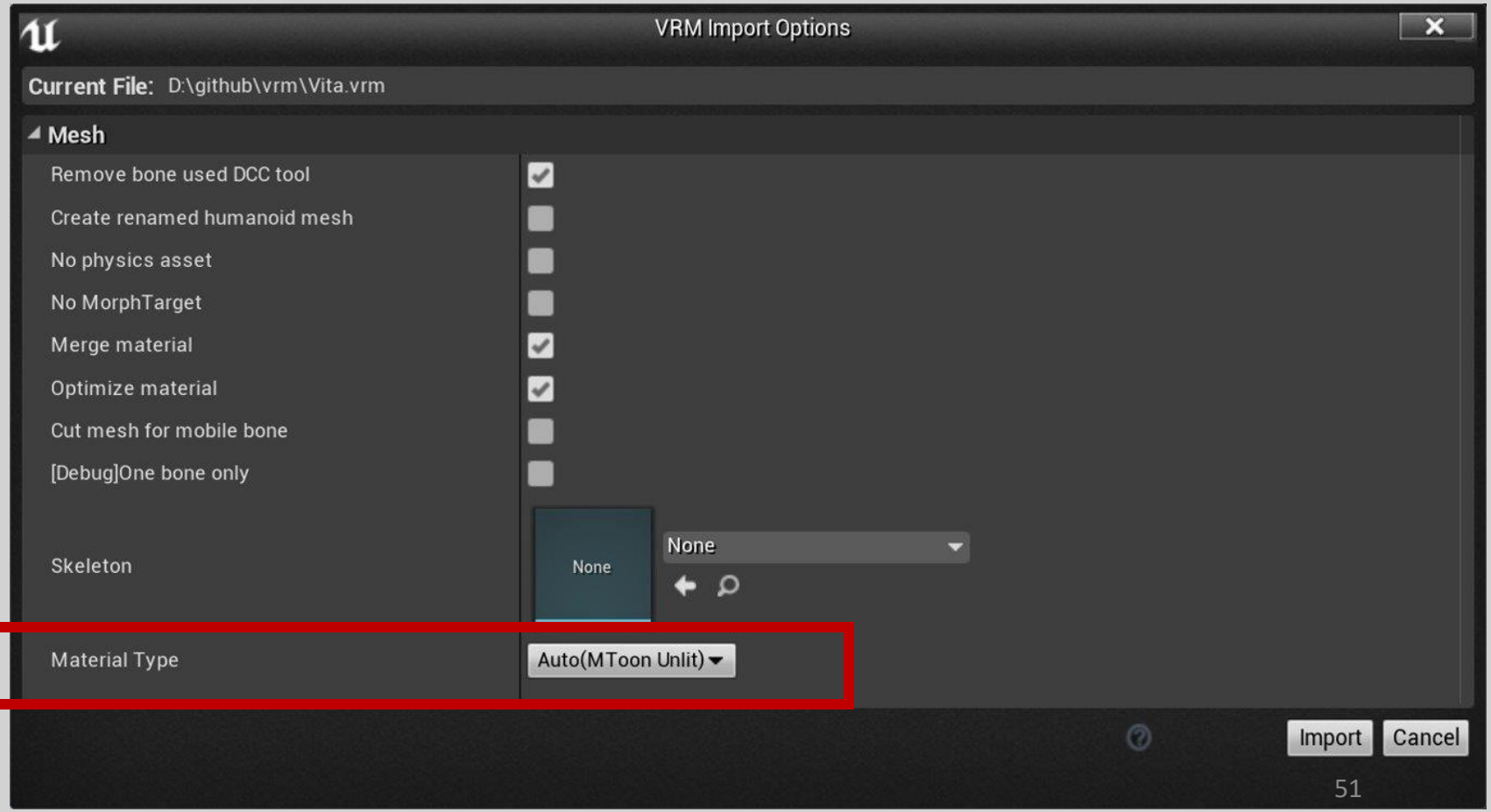
# Lit版マテリアル

- BaseColorを利用すれば、全てのライトの影響を受けることができる
  - ただし陰影の影響も大きくなり、トゥーンらしさは減る

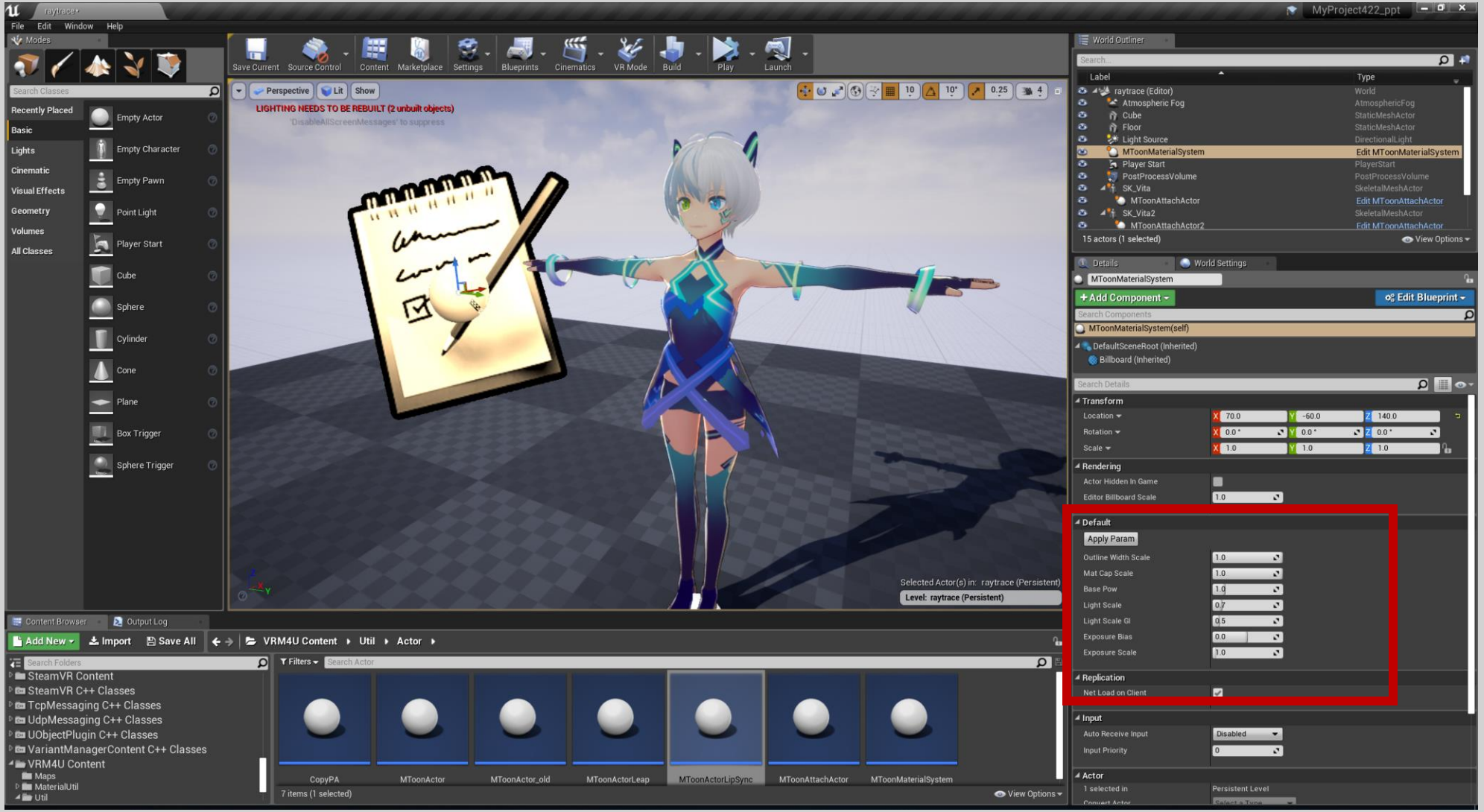
- Litにすれば、SSAOやレイトレーシングを反映できる
  - ただしSSAOは意図しない箇所（特に顔）に黒が乗る。見栄えが悪い。
    やるならレイトレAO

# VRM4UはLit/Unlit選べます

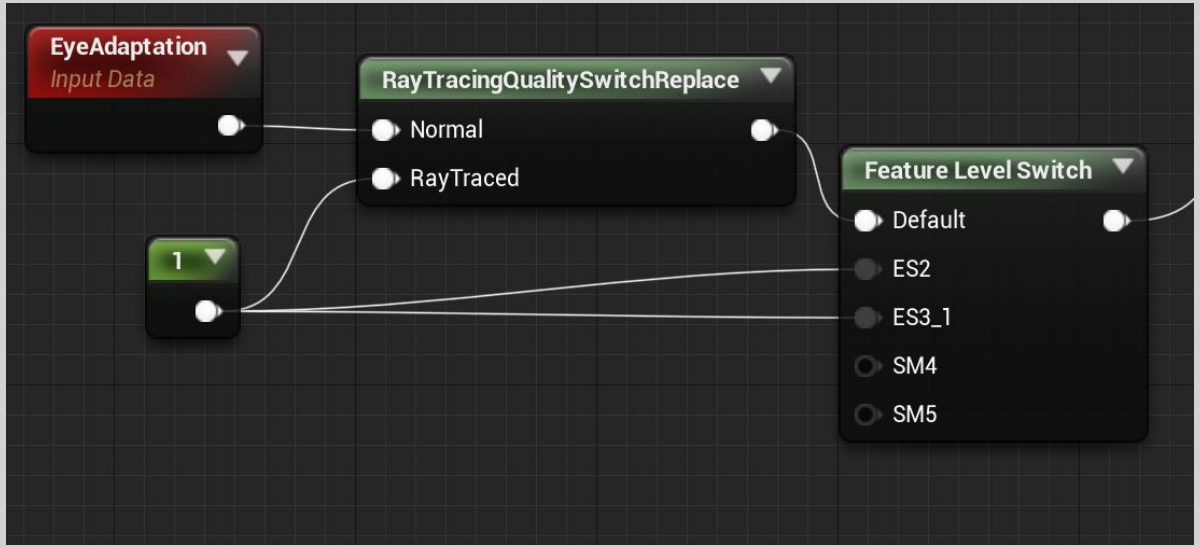
- デフォルトはUnlit



# シェーダパラメータ調整



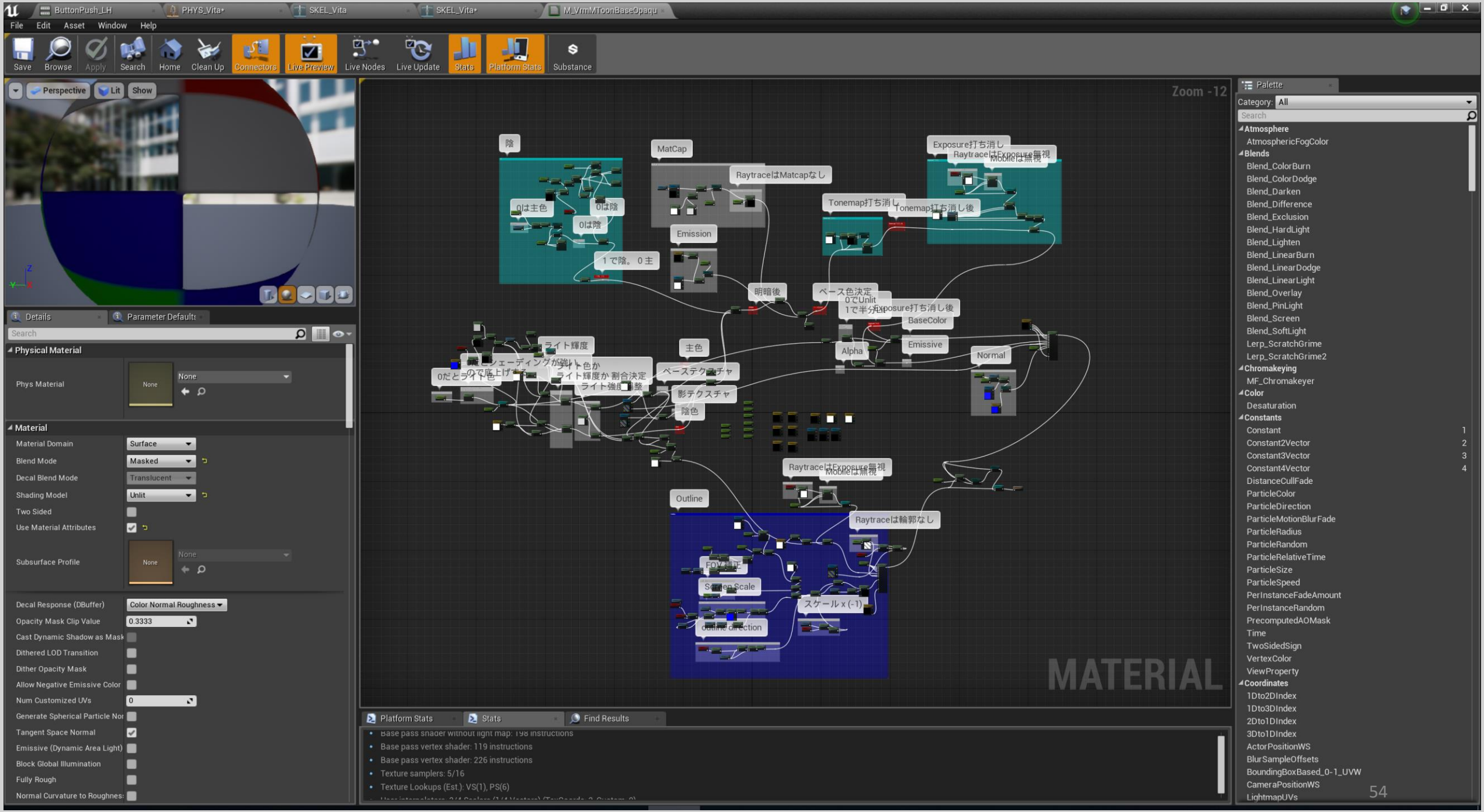
# モバイル、レイトレの対応





EyeAdaptationノードは
レイトレ中とモバイルで使えない

UE4.21以前における、
RayTrace判別ノード


ButtonPush_LH
PHYS_Vita*
SKEL_Vita
SKEL_Vita*
M_VrmMToonBaseOpaqu
File
Edit
Asset
Window
Help
Save
Browse
Apply
Search
Home
Clean Up
Connectors
Live Preview
Live Nodes
Live Update
Stats
Platform Stats
Substance
Perspective
Lit
Show
Details
Parameter Defaults
Search
Physical Material
Phys Material
None
Material
Material Domain
Surface
Blend Mode
Masked
Decal Blend Mode
Translucent
Shading Model
Unlit
Two Sided
Use Material Attributes
Subsurface Profile
Decal Response (DBuffer)
Color Normal Roughness
Opacity Mask Clip Value
0.3333
Cast Dynamic Shadow as Mask
Dithered LOD Transition
Dither Opacity Mask
Allow Negative Emissive Color
Num Customized UVs
0
Generate Spherical Particle Nor
Tangent Space Normal
Emissive (Dynamic Area Light)
Block Global Illumination
Fully Rough
Normal Curvature to Roughnes:
Zoom -12
陰
MatCap
RaytraceはMatcapなし
Exposure打ち消し
RaytraceはExposure無視
Mobileは無視
Tonemap打ち消し
Tonemap打ち消し後
0は主色
0は陰
0は陰
Emission
1で陰。0主
明暗後
ベース色決定
0でUnlit
1で半分以
Exposure打ち消し後
BaseColor
Emissive
Alpha
Normal
ライト輝度
主色
ライト色か
ライト輝度か 割合決定
ライト強度調整
ベーステクスチャ
影テクスチャ
陰色
0だとライト色
RaytraceはExposure無視
Mobileは無視
Outline
Raytraceは輪郭なし
FOV補正
Screen Scale
スケール x (-1)
outline direction
MATERIAL
Palette
Category:
All
Search
Atmosphere
AtmosphericFogColor
Blends
Blend_ColorBurn
Blend_ColorDodge
Blend_Darken
Blend_Difference
Blend_Exclusion
Blend_HardLight
Blend_Lighten
Blend_LinearBurn
Blend_LinearDodge
Blend_LinearLight
Blend_Overlay
Blend_PinLight
Blend_Screen
Blend_SoftLight
Lerp_ScratchGrime
Lerp_ScratchGrime2
Chromakeying
MF_Chromakeyer
Color
Desaturation
Constants
Constant
Constant2Vector
Constant3Vector
Constant4Vector
DistanceCullFade
ParticleColor
ParticleDirection
ParticleMotionBlurFade
ParticleRadius
ParticleRandom
ParticleRelativeTime
ParticleSize
ParticleSpeed
PerInstanceFadeAmount
PerInstanceRandom
PrecomputedAOMask
Time
TwoSidedSign
VertexColor
ViewProperty
Coordinates
1Dto2DIndex
1Dto3DIndex
2Dto1DIndex
3Dto1DIndex
ActorPositionWS
BlurSampleOffsets
BoundingBoxBased_0-1_UVW
CameraPositionWS
LightmapUVs
Platform Stats
Stats
Find Results
Base pass shader without light map: 198 instructions
Base pass vertex shader: 119 instructions
Base pass vertex shader: 226 instructions
Texture samplers: 5/16
Texture Lookups (Est.): VS(1), PS(6)

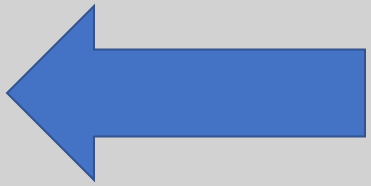
# マテリアル　VRM4Uで見えてきたこと

- トゥーン以外の、様々なNPRな表現が可能
  - 完全な色のコントロール
  - 影、陰影の判別
  - PoseableMeshを利用した複数パス描画

- NPRとレイトレースを組み合わせた、ちょっと豪華な絵を出せる

# アジェンダ

- VRM4Uの方針
- マテリアル
- インポーター
- アニメーション
- モバイル対応
- まとめ



# インポーター

# インポーター　目標

- VRM(glTF)が読めればOK！

- 制約
  - VRMの思想に沿うのであれば、、
    - アバターなので、ゲーム実行中でのランタイムロードは必須。
    - 調整済のファイルなので、インポートしてそのまま使う前提。
      後工程による手動調整は原則無し。

# ランタイムロードとは

インポート

- エディタ操作中に使用する
- コンテンツブラウザにVRMをDrag&Drop

- Editorモジュール

ランタイムロード

- ゲーム実行中に使用する
- ゲームウィンドウにVRMをDrag&Drop
  - フルパスから読めればOK

- Runtimeモジュール

あくまで このスライド内での用語です

# インポーターの課題

UE4にモデルをラインタイムロードする機能はない

# インポーターの課題　気持ち的な面

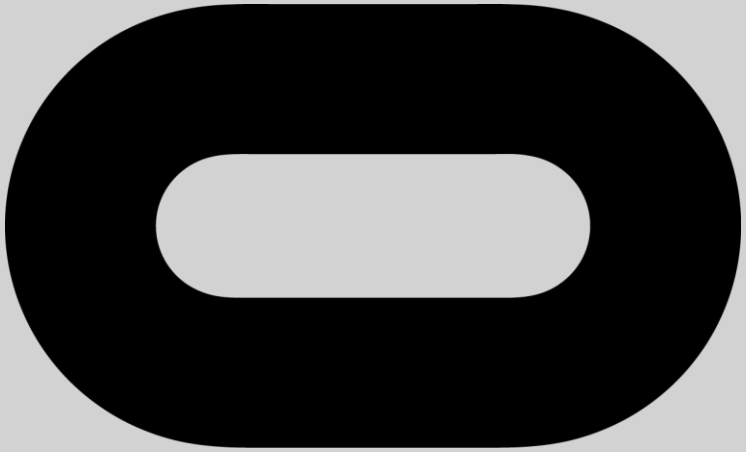
- glTFローダ、作るのは大変
  しかもC++で。

- UE4がモデルをランタイムロードできるのか不明
  しかもエンジン改造なし、プラグインでの実装

- UE4がバージョンアップしたら、ビルドが通らなくなる予感がする

# インポーター　やる気が出るまでの経緯

- VR、VTuberが盛り上がってて羨ましいなぁ　という気持ち



# assimpを使ってテスト実装することを決意

- OSSのglTFローダ

- 決め手
  - 不安材料はあるが、glTF読み込み周りのソースは とてもキレイ
    - 当時(2018/10月)はskinmeshに未対応だった。自前で追加実装することを決意
    - 現在はassimp本家側で対応済
  - コンバート機能がある
    - データをFBX出力することで、単体でglTFロードチェックができる
  - UE4プラグインであるRuntimeMeshLoaderで使っている人がいる
    - このプラグインはモデルをProceduralMeshComponentに変換するだけ。

# ランタイムローダの作成

- アプローチ
  - NewObject<USkeletalMesh>() でアセットを生成、Spawnさせて、
    停止する箇所に 片っ端からデータを埋め込んでいく

# ランタイムローダの作成

- アプローチ
  - NewObject<USkeletalMesh>() でアセットを生成、Spawnさせて、
    停止する箇所に 片っ端からデータを埋め込んでいく

気合

# ランタイムローダの作成 要所

- SkeletalMesh->GetResourceForRendering()
  ->LODRenderData[0]
  - StaticVertexBuffers
    - 頂点情報。最低限これは埋める。
  - MultiSizeIndexContainer
    - 描画用の頂点情報。ランタイムロードするなら、これも埋める
    - GameThreadからの書き換えで停止するメンバは、RenderThreadから書き換える

- SkeletalMesh->GetImportedModel()->LODModels[0].Sections;
  - メッシュ、マテリアルの情報。これも埋める

- SkeletalMesh->RegisterMorphTarget()
  - ブレンドシェイプ情報。これも埋める

- UE4の頂点ウェイトはuint8なので、端数が出る。最後に正規化する

# VRM（glTF）取扱時の注意（1/2）

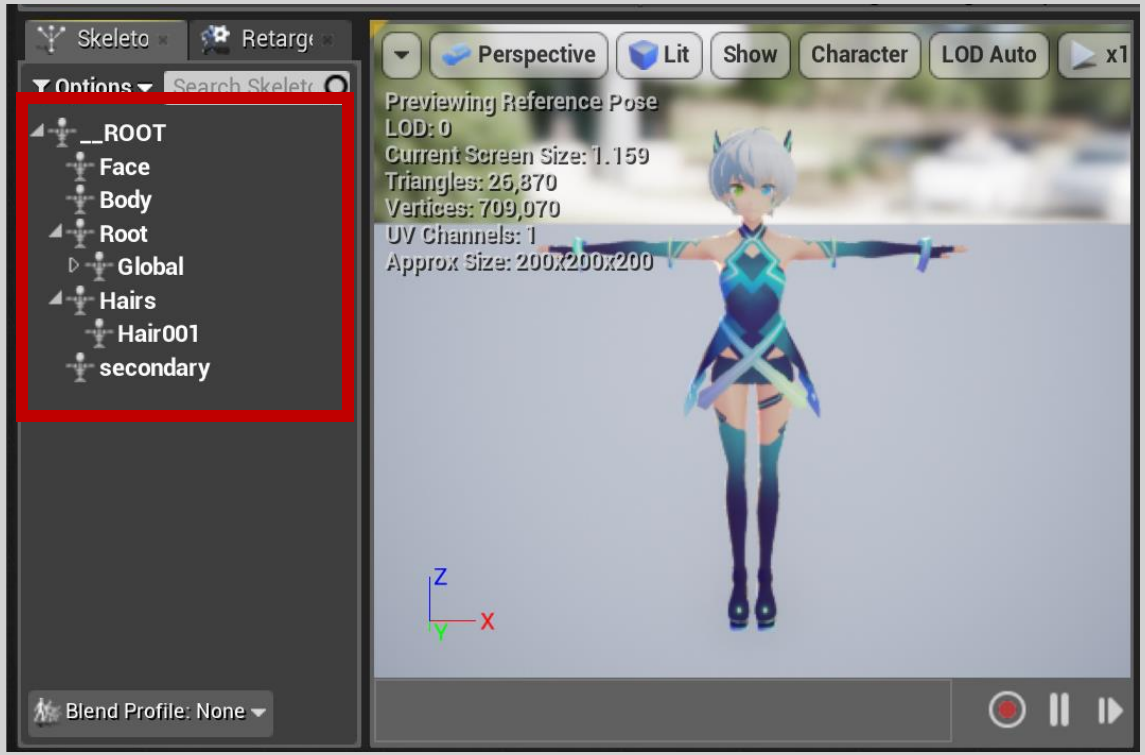
- 座標系を変更する
  - Z-upに変換
  - スケールをUnrealUnitへ変換(x100)

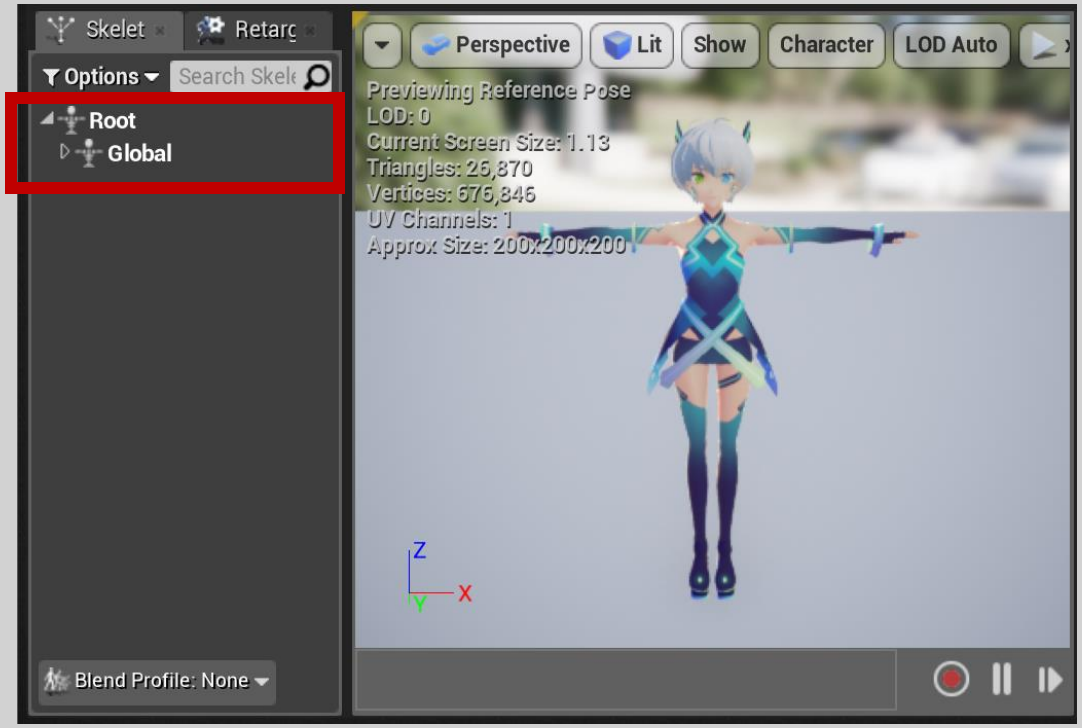
- 複数Root骨を対応する
  - 不要な骨の削除
  - 削除しない場合は、ダミーRoot骨の追加

# VRM（glTF）取扱時の注意（2/2）

- アセット名として利用禁止な文字のチェック
  - INVALID_OBJECTNAME_CHARACTERSと、INVALID_LONGPACKAGE_CHARACTERS
  - 使用例はFName::IsValidXName() で検索
- 文字コードの変換
  - VRMパラメータは UTF8_TO_TCHAR で変換
    - 内部的にはJSONで、文字コードはUTF8になっている
- 特にテクスチャ、マテリアル名に、
  日本語や禁止文字が入っていることが多い

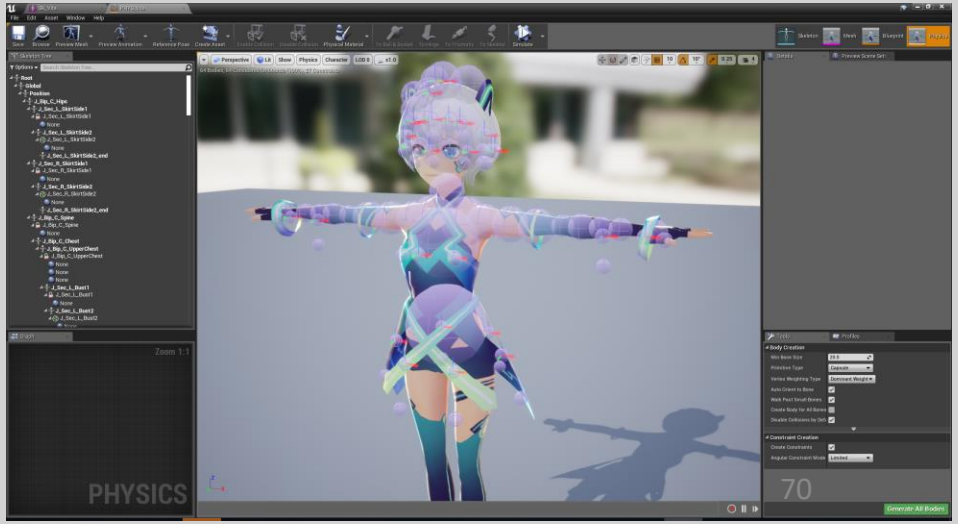
# 複数Root骨の対応


Skeleto
Retarge
Options
Search Skeleto
__ROOT
Face
Body
Root
Global
Hairs
Hair001
secondary
Perspective
Lit
Show
Character
LOD Auto
x1
Previewing Reference Pose
LOD: 0
Current Screen Size: 1.159
Triangles: 25,870
Vertices: 709,070
UV Channels: 1
Approx Size: 200x200x200
Z
X
Y
Blend Profile: None


Skelet
Retarg
Options
Search Skele
Root
Global
Perspective
Lit
Show
Character
LOD Auto
Previewing Reference Pose
LOD: 0
Current Screen Size: 1.13
Triangles: 25,870
Vertices: 676,846
UV Channels: 1
Approx Size: 200x200x200
Z
X
Y
Blend Profile: None

# PhysicsAssetの作成

- コリジョン、コンストレイントを生成して、パラメータを埋めればOK
  - bs = NewObject<USkeletalBodySetup>();
  - ct = NewObject<UPhysicsConstraintTemplate>();
  - physicsAsset->SkeletalBodySetups.Add(bs);
  - physicsAsset->ConstraintSetup.Add(ct);

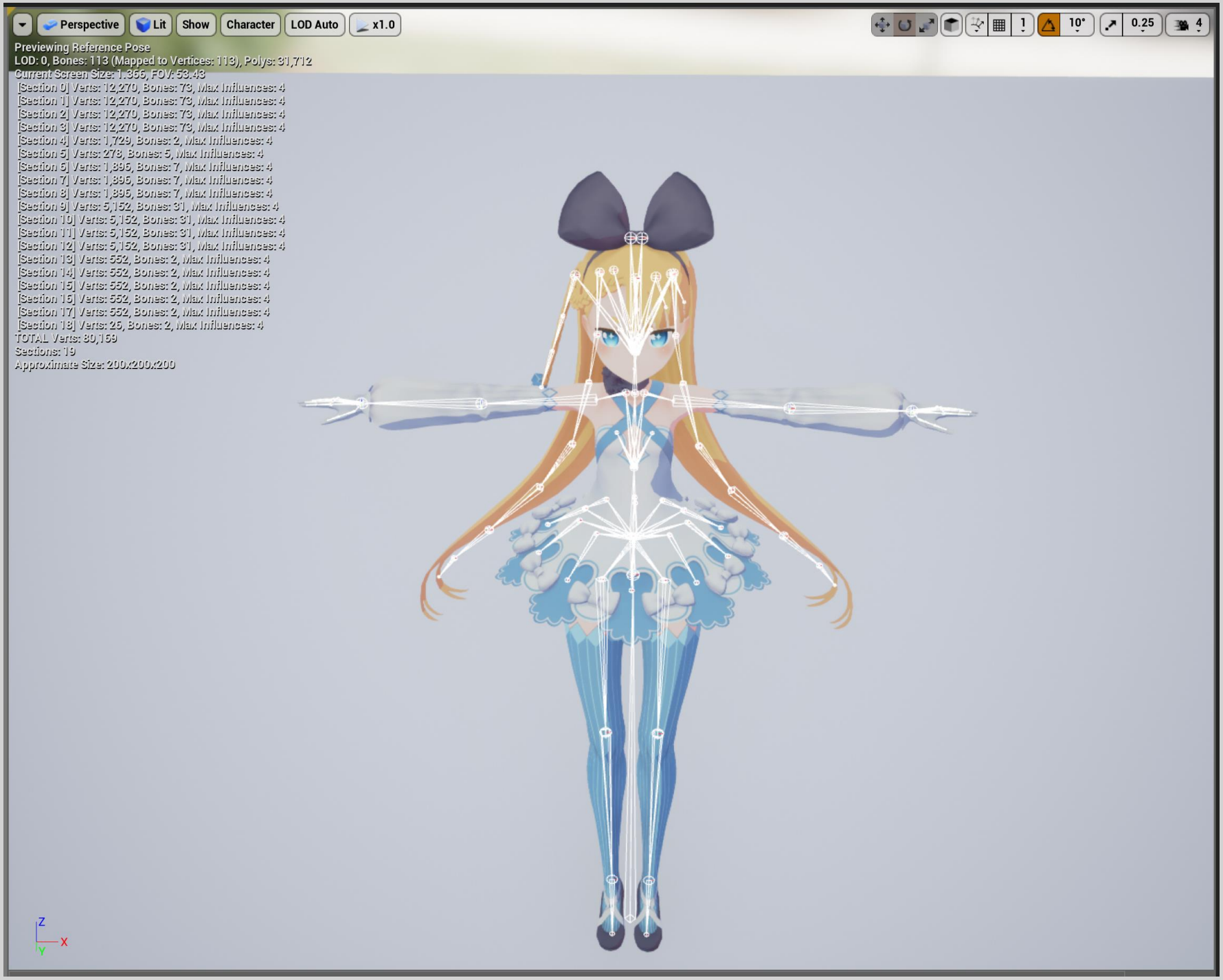


# インポーター つづきます

# 揺れ骨　飽くなき欲求

- 骨が暴れないようにしたい
  - 物理が荒ぶってしまう。コリジョンが激しくめりこんだ時が顕著。
  - 計算回数を増やせば、ある程度は改善される。

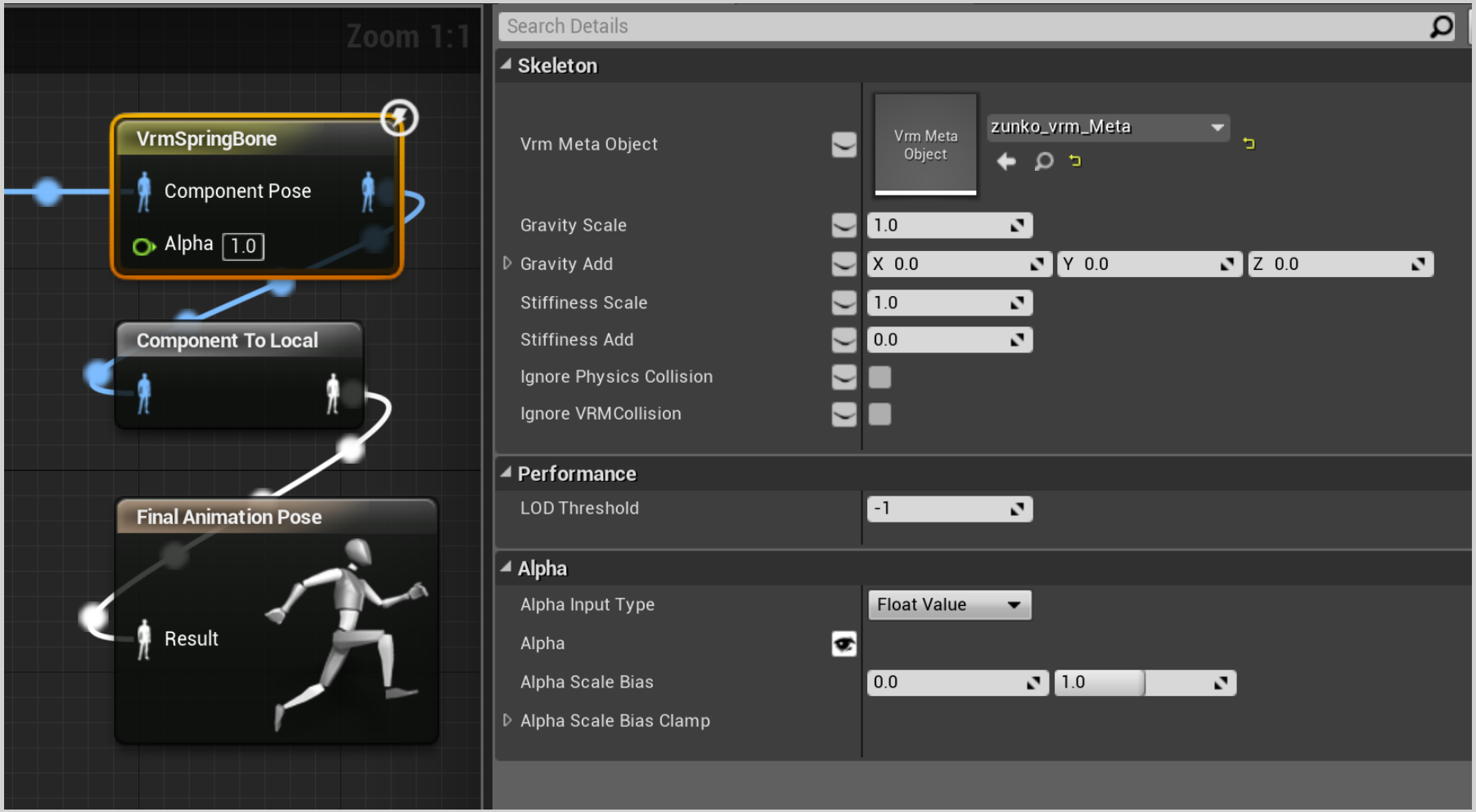
- 意図した形状を保持したい
  - 髪の毛が重力で垂れ下がってしまう。
  - 可動範囲指定もできるが、動きが乱暴。雑。


Perspective
Lit
Show
Character
LOD Auto
x1.0
10°
0.25
4
Previewing Reference Pose
LOD: 0, Bones: 113 (Mapped to Vertices: 113), Polys: 31,712
Current Screen Size: 1.366, FOV: 53.43
[Section 0] Verts: 12,270, Bones: 73, Max Influences: 4
[Section 1] Verts: 12,270, Bones: 73, Max Influences: 4
[Section 2] Verts: 12,270, Bones: 73, Max Influences: 4
[Section 3] Verts: 12,270, Bones: 73, Max Influences: 4
[Section 4] Verts: 1,729, Bones: 2, Max Influences: 4
[Section 5] Verts: 278, Bones: 5, Max Influences: 4
[Section 6] Verts: 1,896, Bones: 7, Max Influences: 4
[Section 7] Verts: 1,896, Bones: 7, Max Influences: 4
[Section 8] Verts: 1,896, Bones: 7, Max Influences: 4
[Section 9] Verts: 5,152, Bones: 31, Max Influences: 4
[Section 10] Verts: 5,152, Bones: 31, Max Influences: 4
[Section 11] Verts: 5,152, Bones: 31, Max Influences: 4
[Section 12] Verts: 5,152, Bones: 31, Max Influences: 4
[Section 13] Verts: 552, Bones: 2, Max Influences: 4
[Section 14] Verts: 552, Bones: 2, Max Influences: 4
[Section 15] Verts: 552, Bones: 2, Max Influences: 4
[Section 16] Verts: 552, Bones: 2, Max Influences: 4
[Section 17] Verts: 552, Bones: 2, Max Influences: 4
[Section 18] Verts: 25, Bones: 2, Max Influences: 4
TOTAL Verts: 80,159
Sections: 19
Approximate Size: 200x200x200
Z
X
Y

# 揺れ骨自前実装への道

- VRMSpringBoneはUnityの実装が公開されている
- 理論上はUE4に移植できるはず
- ソース（VRMSpringBone.cs）は たったの329行
  - まぁ… いけるかなぁ…

# VRMSpringBoneノード作りました

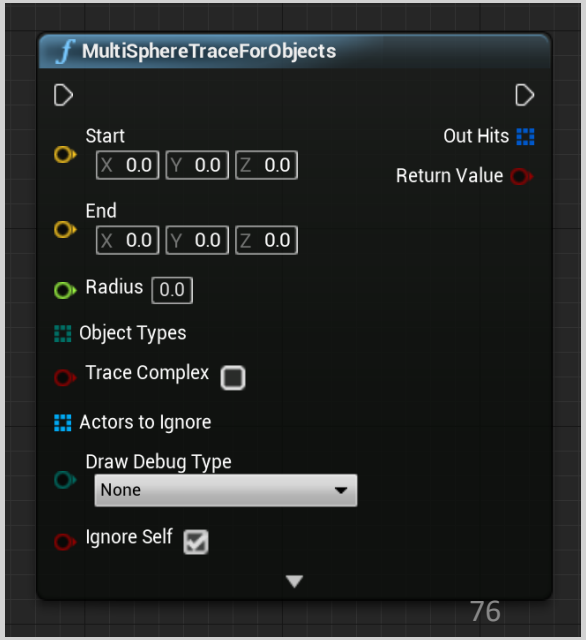


# VRMSpringBone実装 の要所

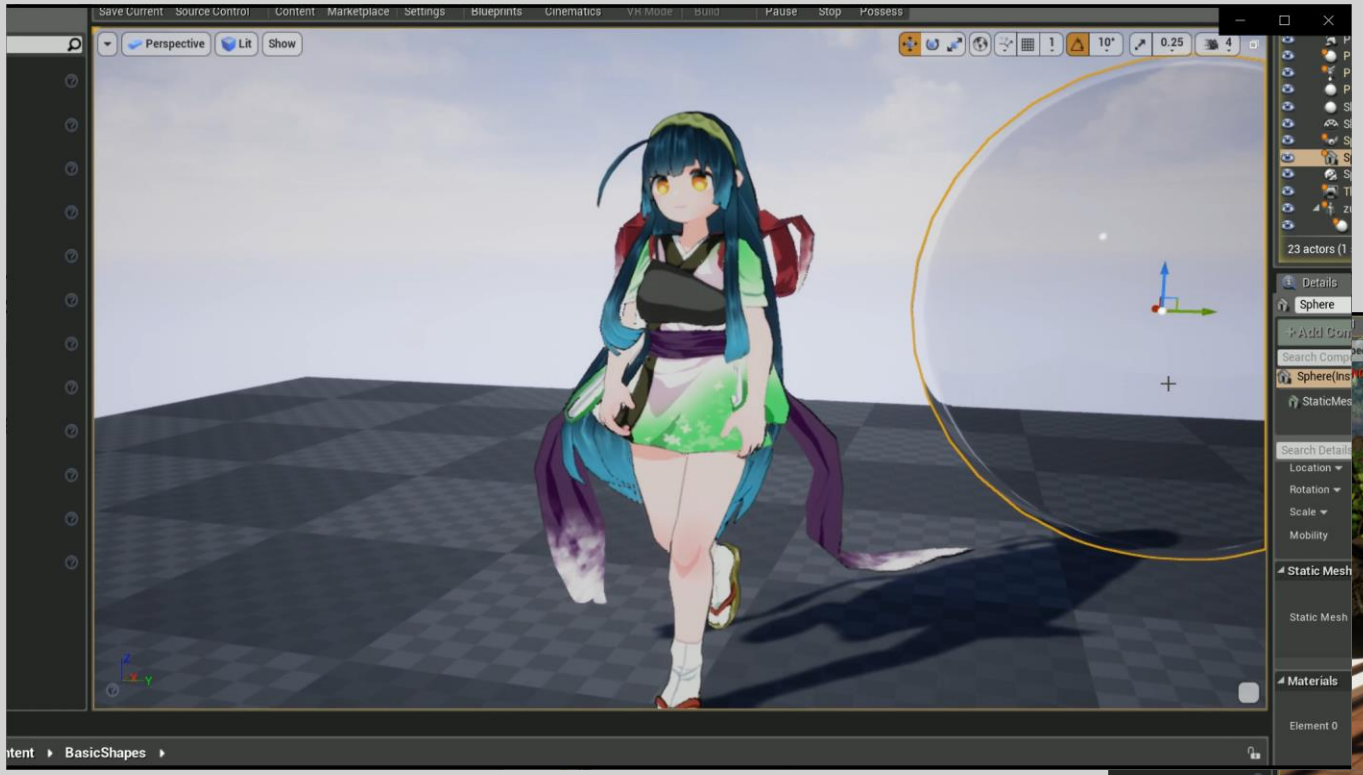
- FAnimNode_ModifyBone を参考に。
  - 各種座標系でのTransformの取得、書き換え方法がわかる

- UE4コリジョンとの衝突をとる
  - ラインとレース系の関数で衝突情報が得られる
  - VRMSpringBone内の衝突計算に統合すればOK！

UKismetSystemLibrary::SphereTraceMulti()



# 動画x2


Save Current
Source Control
Content
Marketplace
Settings
Blueprints
Cinematics
Pause
Stop
Possess
Perspective
Lit
Show
23 actors (1
Details
Sphere
Location
Rotation
Scale
Mobility
Static Mesh
Materials
Element 0
BasicShapes


Lit
Show
971 actors (2 selected)
Details
phys
Physics
Simulate Physics
Linear Damping
Angular Damping
Enable Gravity
Constraints
Lock Position
Lock Rotation
Mode
Ignore Radial Impulse
Ignore Radial Force
Center Of Mass Offset
Mass Scale
Sleep Family
Generate Wake Events
NewFolder4


Zoom 1:1
VrmSpringBone
Component Pose
Alpha 1.0
Component To Local
Final Animation Pose
Result
Search Details
Skeleton
Vrm Meta Object
Vrm Meta Object
zunko_vrm_Meta
Gravity Scale
1.0
Gravity Add
X 0.0
Y 0.0
Z 0.0
Stiffiness Scale
1.0
Stiffiness Add
0.0
Ignore Physics Collision
Ignore VRMCollision
Performance
Alpha Scale Bias
0.0
1.0
Alpha Scale Bias Clamp
揺れ具合を調整できる。
動きが激しいアクションゲーム向け
外力を設定することで、風でなびくような効果も出せる

# インポーター　まとめ

- VRMを読めるようになった
  - インポート/ランタイムロードできるようになった
  - ブレンドシェイプ、コリジョンも読めるようになった

- VRMSpringBoneを再現できるようになった
  - UE4のコリジョンと干渉できるようになった

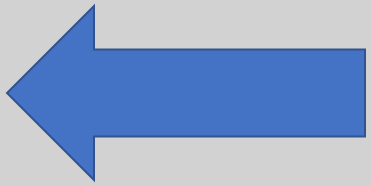
# インポーター　VRM4Uで見えてきたこと

- StaticMeshのランタイムロードも、おそらく可能

- カスタム揺れ骨ノードは作れる
  - PhysicsAssetやAnimDynamicsノードの挙動に不満がある人向け
  - 思い切って、作ってみてはいかが？

# アジェンダ

- VRM4Uの方針
- マテリアル
- インポーター
- アニメーション
- モバイル対応
- まとめ



# アニメーション

# アニメーション　目標

- ロードしたモデルに、アニメーションを適用できればOK！

- 制約
  - ランタイムロードしたモデルも、アニメーションを適用できる

# アニメーションの課題

UE4にランタイムでリターゲットする機能はない

# 既存のアニメーションを適用するには

- （A）Skeletonを共通化する
- （B）アニメーションアセットをリターゲットする

# 既存のアニメーションを適用するには

- （A）Skeletonを共通化する

骨階層が異なったり、
同名の骨が別階層にあるとエラー

- （B）アニメーションアセットをリターゲットする

両方とも、Editorの機能。
ランタイムでは利用できない。

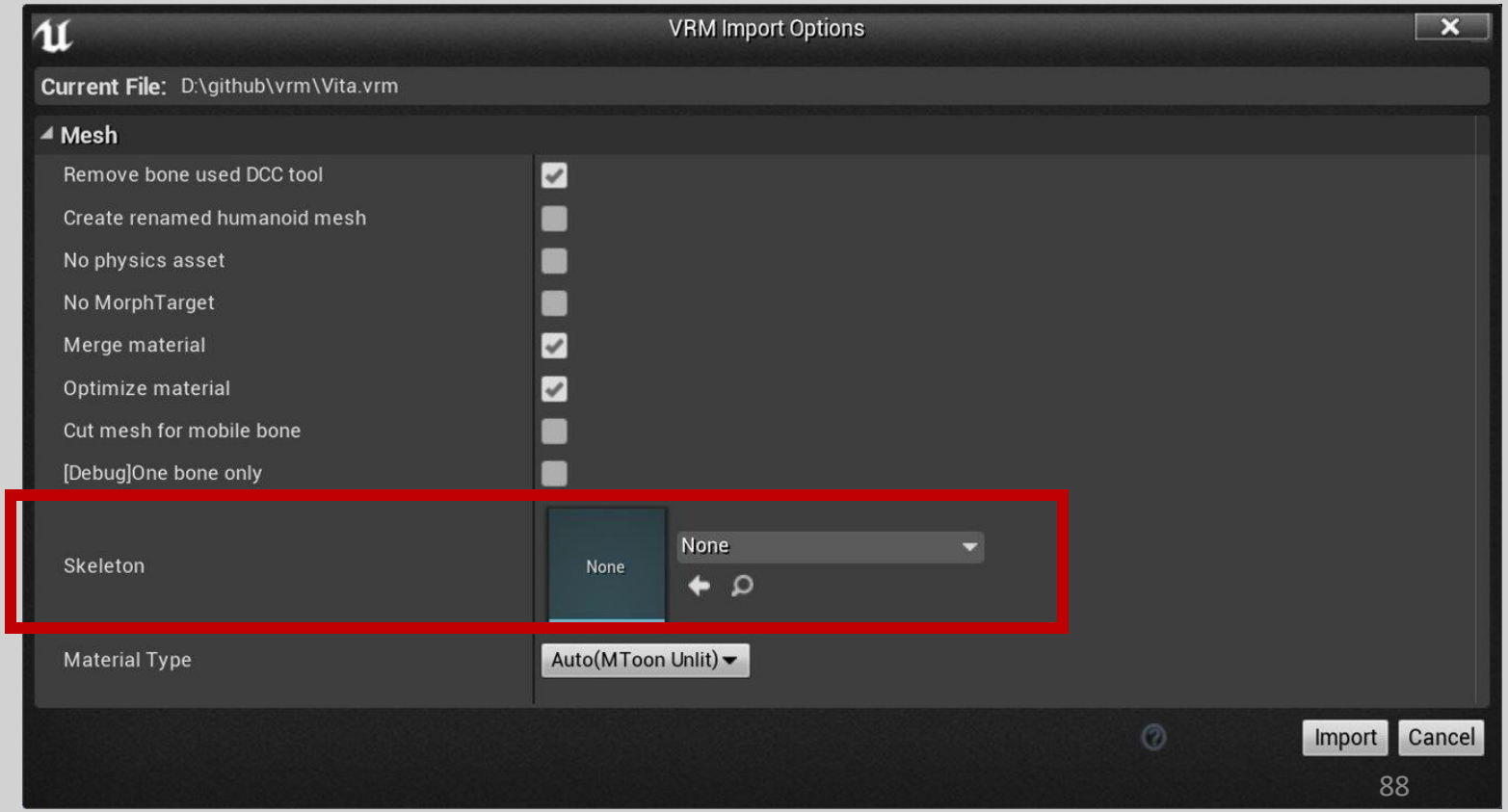
# 既存のアニメーションを適用するには

- （A）Skeletonを共通化する
- （B）アニメーションアセットをリターゲットする
- （C）ランタイムリターゲットする

NEW

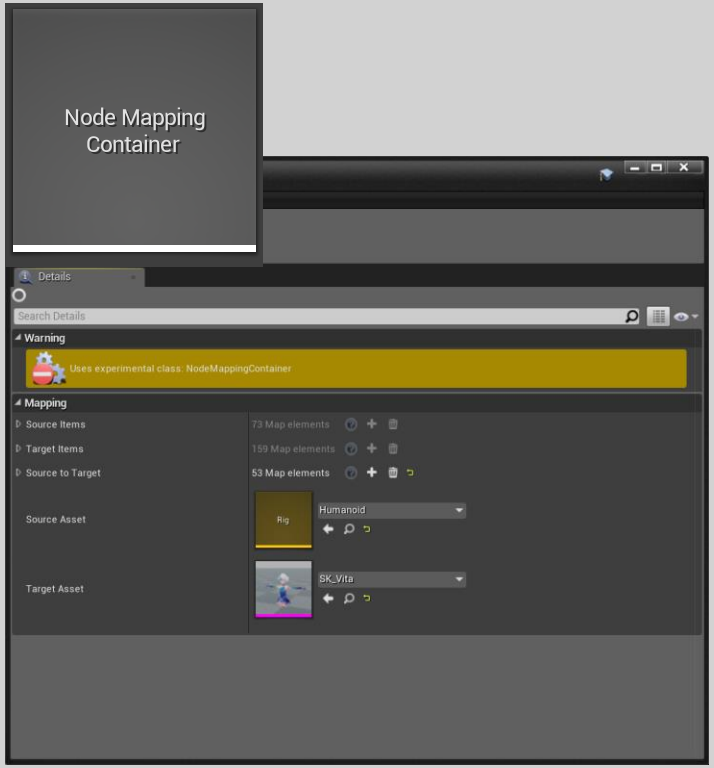
# （A）Skeletonを共通化する

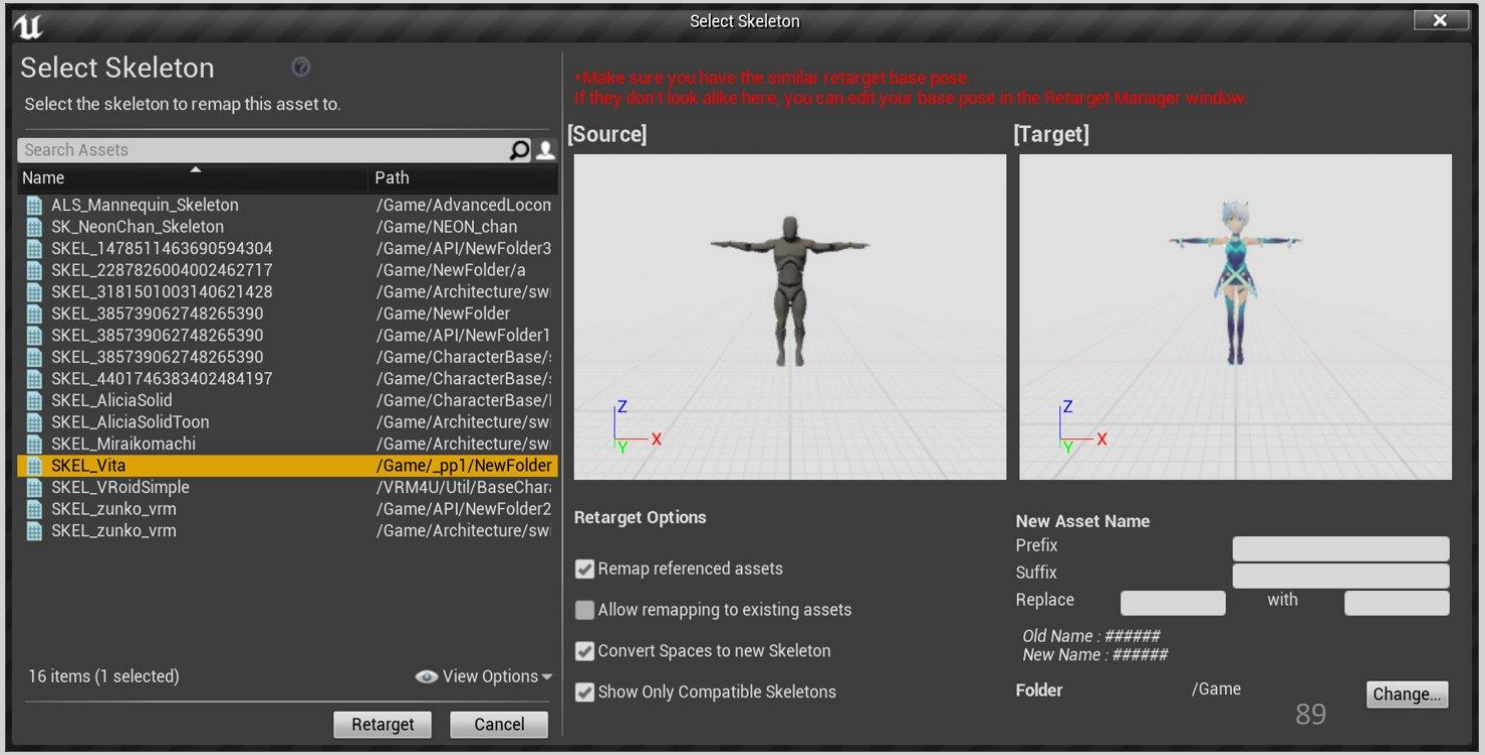
- VRMインポート時にSkeletonをセットすればOK
- FBXインポートと同じ



# （B）アニメーションをリターゲット

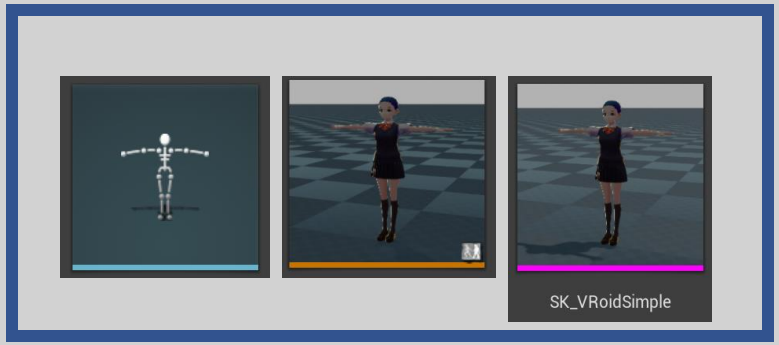
- AnimAssetをリターゲットする
- Humanoidに対応するNodeMappingContainerがあればOK



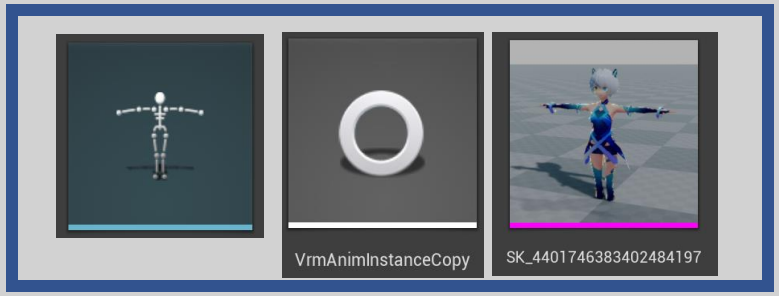



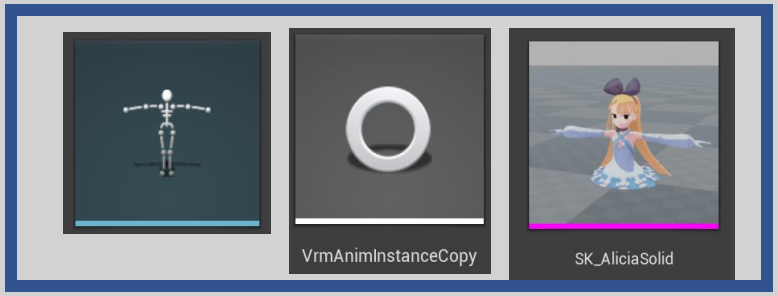
# （C）ランタイムリターゲット

- AnimInstanceにコピーアセットを設定すればOK




コピー元となるメッシュ。
エディタで作成したAnimBPをセット




コピー先となるメッシュ。
VrmAnimInstanceCopyをセット

# アニメーション共通化 実装のポイント

- (A)Skeletonの共通化
  - 実装 : Skeleton->MergeAllBonesToBoneTree を呼んで成功すればOK
  - 注意:骨の階層が異なっていたり、別階層に同名の骨があるとNG
- (B)アニメーションをリターゲット
  - 実装:Humanoidに対応するNodeMappingContainerを出力すればOK
  - 注意:元データはT-poseにする
- (C)ランタイムでリターゲット
  - 実装:Humanoidで対応する骨の姿勢をコピーすればOK
  - 注意:骨の回転成分のみコピーする。移動はコピーしない。

# ランタイムリターゲット　残る課題

- 手の大きさ、長さが異なることによる問題
  - オブジェクトを掴んだり、両手を合わせたりすると、位置がずれる

- 足の長さが異なることによる問題
  - 足が地面にめり込む、浮く
  - 足が滑る、接地感がなくなる

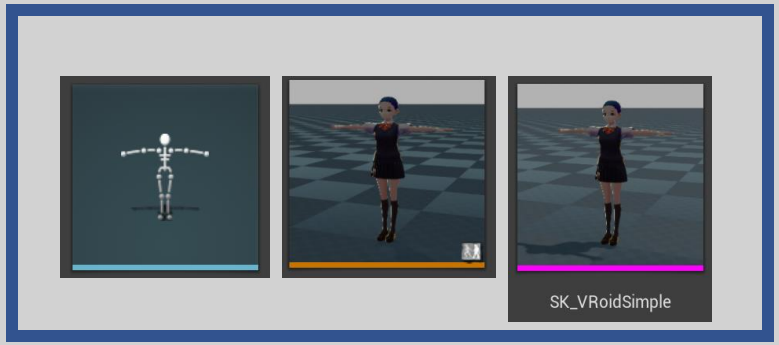
- リターゲット元を工夫すれば軽減は可能…？

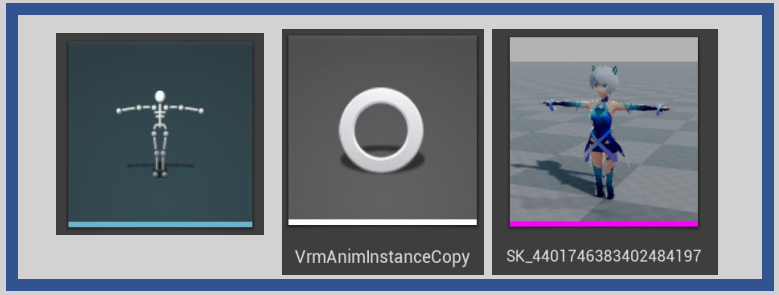
# アニメーション つづきます

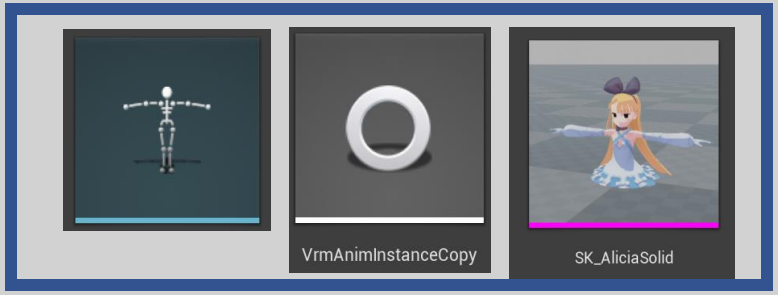
# リターゲットへ　飽くなき欲求

- Humanoid骨はリターゲットしつつ、
  揺れ骨はVRMSpringBoneを使いたい
  - リターゲット先はAnimInstanceを利用しており、AnimBPが無い。
    つまりVRMSpringBoneノードを利用できない

# (C)ランタイムリターゲット

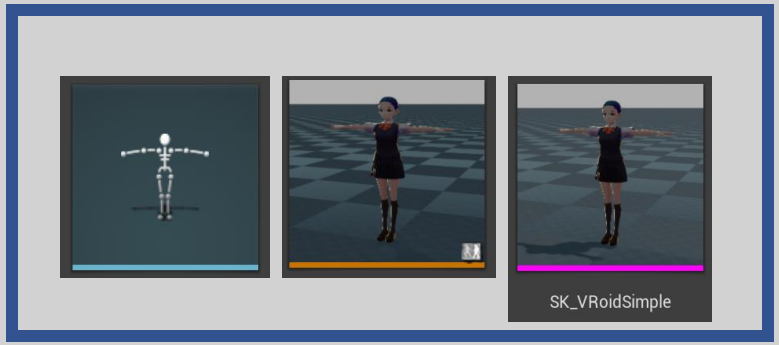


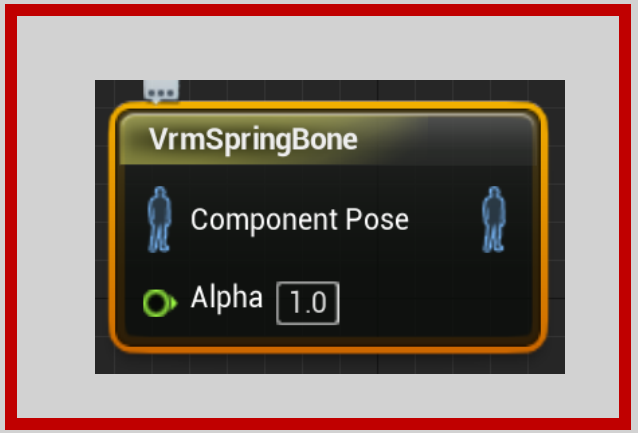



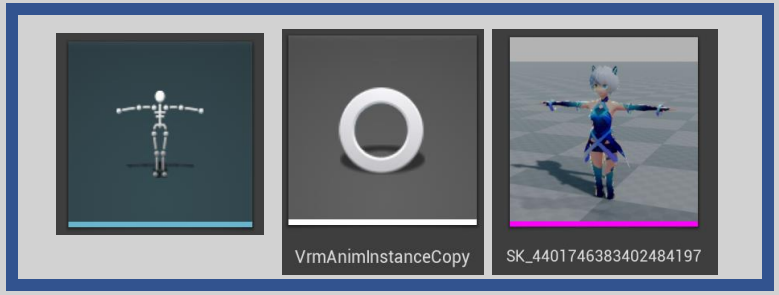



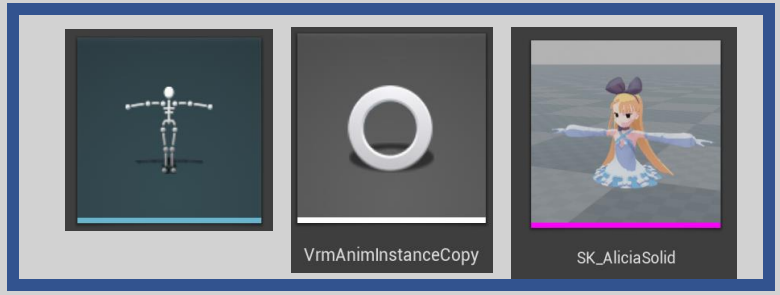
# (C)ランタイムリターゲット

ノードをどこで設定するの?



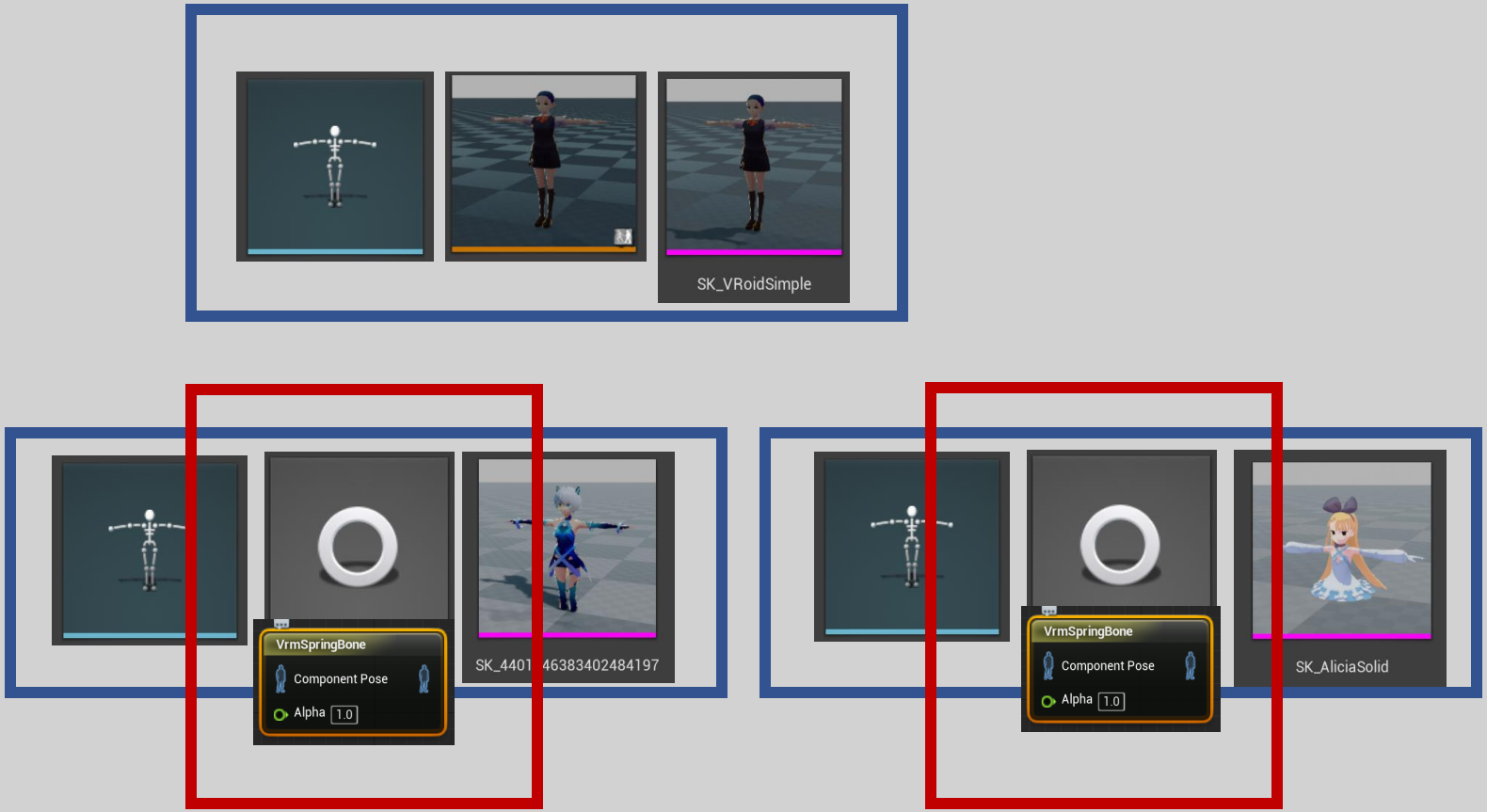



# (C)ランタイムリターゲット

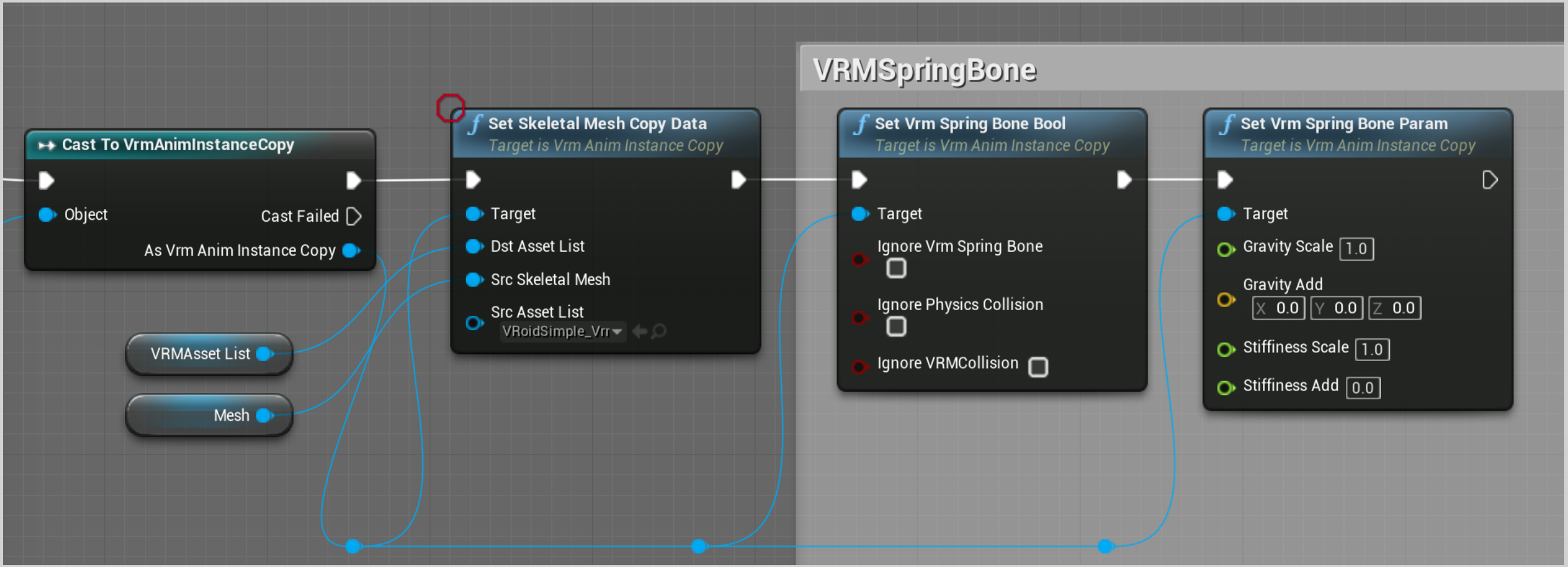
- つまり、C++からVrmSpringBoneノードを呼び出せばOK！



# AnimInstanceからのノード呼び出し 要所

- ノードをnewして、
  - node = MakeShareable(new FAnimNode_VrmSpringBone());
- 初期化して、
  - node->Initialize_AnyThread(InitContext);
  - node->ComponentPose.SetLinkNode(node);
- ノードの計算を呼び出し、それを自分に書き戻せばOK
  - InputCSPose.Pose.InitPose(Output.Pose);
  - Node->EvaluateComponentSpace_AnyThread(InputCSPose);
  - ConvertToLocalPoses(InputCSPose.Pose, Output.Pose);

# AnimInstanceからノードにアクセス



# アニメーション　まとめ

- エディタのリターゲット機能を使うことができた。

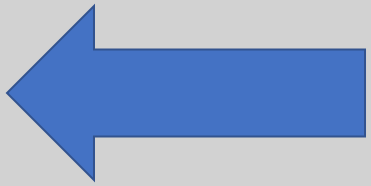
- ランタイムでリターゲットできるようになった。
  - VRMSpringBoneも適用できるようになった

# アニメーション　VRM4Uで見えてきたこと

- 後からSkeletonをマージすることが可能
  - リインポートしなくても、マージのみ処理できるはず
- PhysicsAssetを別Skeletonにコピー可能
  - 一部であればVRM4Uで対応済
- より汎用的なランタイムリターゲットが可能？
  - RIGを利用すれば、おそらく可能
  - ただしRIGはGameビルドでは利用できない

# アジェンダ

- VRM4Uの方針
- マテリアル
- インポーター
- アニメーション
- モバイル対応
- まとめ



# モバイル

# モバイル　目標

- PC向けと同じくらい手軽に利用できる

- 制約
  - キレイなトゥーンが描画できている

# モバイルの課題

メッシュが参照する骨数が75を越えると停止する

※骨数自体は75を越えてもOK

# モバイルの課題　対処

- ソースで最大数を書き換える

- メッシュを分割する

# モバイルの課題　対処

- ソースで最大数を書き換える

MAX_GPU_BONE_MATRICES_UNIFORMBUFFER = 75
ビルドの時間がかかりすぎる。ローカルだと数時間…

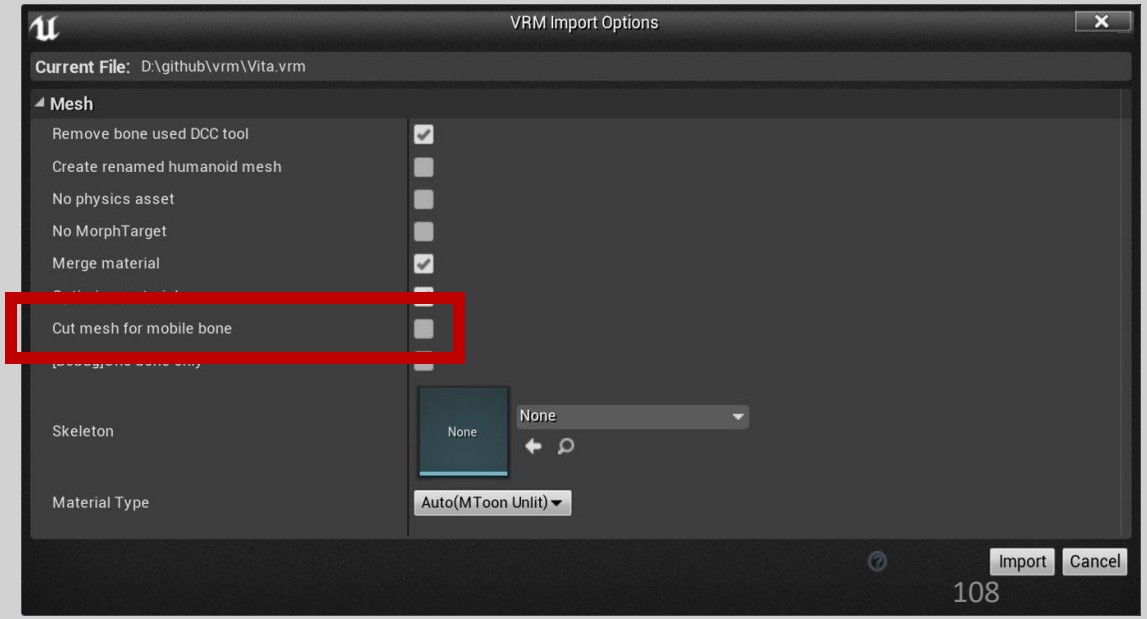
- メッシュを分割する

骨をカウントしながら分割する実装がややこしい
私は挫けました…

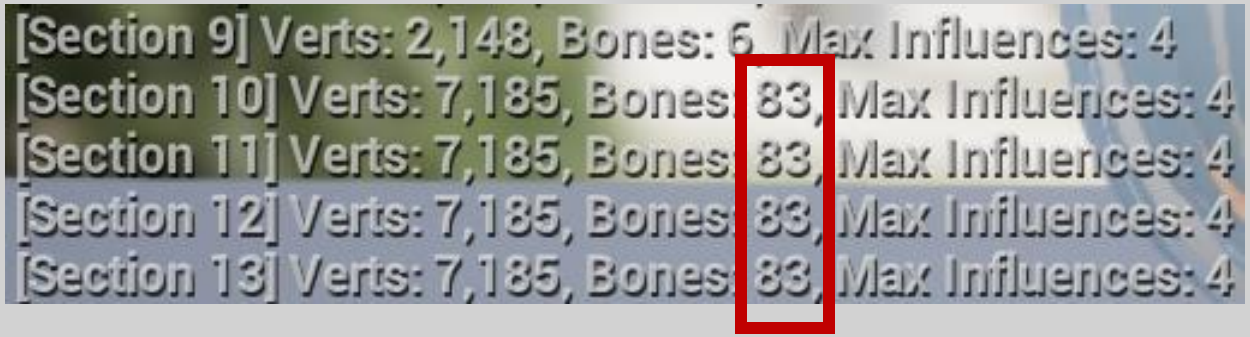
- BoneMapのリダクション（VRM4U）
  - すこし変形してしまう

# BoneMapのリダクション 要所

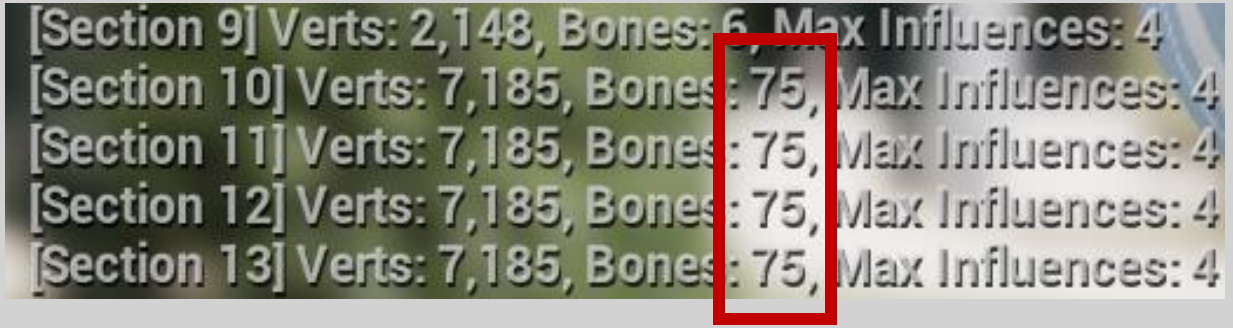
- FSkeletalMeshLODRenderData::RenderSections[n].BoneMap の配列サイズを75以下にすればOK
- 総Weightが小さい（＝影響度が少ない）骨から順に、親骨にWeightを振り直す。



リダクション前

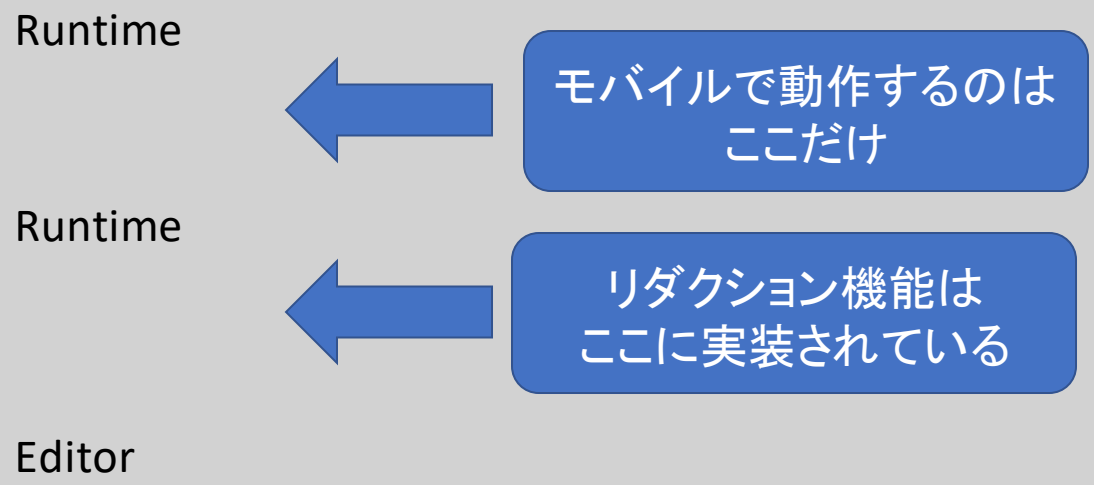




リダクション後

# モバイル対応　モジュール構成

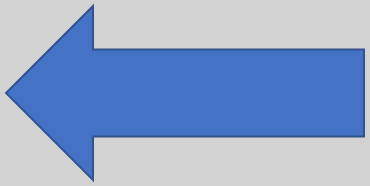
- VRM4U
  - 揺れ骨、リターゲット、アセット定義
- VRM4ULoader
  - VRMインポート、ランタイムロード
- VRM4UImporter
  - インポートUI



# モバイル　XRなんでも来い！


# アジェンダ

- VRM4Uの方針
- マテリアル
- インポーター
- アニメーション
- モバイル対応
- まとめ



# まとめ

# ▶冒頭の動画をもう一度


File Edit Window Help
Modes
Save Current
Source Control
Content
Marketplace
Settings
Blueprints
Cinematics
Build
Compile
Pause
Stop
Possess
Viewport 1
Perspective
Lit
Show
LIGHTING NEEDS TO BE REBUILT (663 unbuilt objects)
Recently Placed
Basic
Lights
Cinematic
Visual Effects
Geometry
Volumes
All Classes
Empty Actor
Empty Character
Empty Pawn
Point Light
Player Start
Cube
Sphere
Cylinder
Cone
Plane
Box Trigger
Sphere Trigger
World Outliner
973 actors (1 selected)
View Options
Details
World Settings
Select an object to view details.
Content Browser
Output Log
Add New
Import
Save All
Content
_pp1
NewFolder4
Texture
Node Mapping Container
Vrm License Object
Vrm Meta Object
Vrm Asset List Object
69 items (1 selected)

おわかりいただけただろうか

# まとめ

- UE4でVRMを扱えそうな気、してきました？

- まずは簡単なプロトタイプやテストプログラム向けにどうぞ
  - VRMのライセンスファイルもインポートされます。是非ご一読ください。

# 良きVRMライフを！

# お借りしたデータ

- VRoid Studio
  - https://studio.vroid.com/
- アリシア・ソリッド
  - https://3d.nicovideo.jp/alicia/
- 東北ずん子
  - https://zunko.jp/

ありがとうございました