Kosmuタスク報告

松田　真

進歩

少々忙しかったため、今回の進捗報告は8月4日（月）から8月7日（木）までの内容となります。

- １日目：UnrealEngine、VS、プロジェクトの方針決定
- ２日目～４日目：開発、デバグ、スライドの準備

１日目

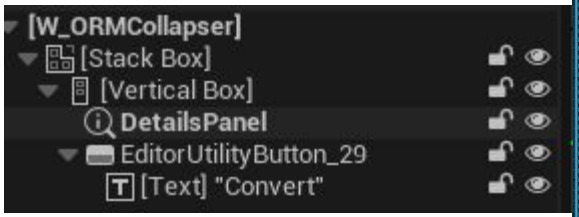
UnrealEngine 5.4、VIsualStudioのインストール

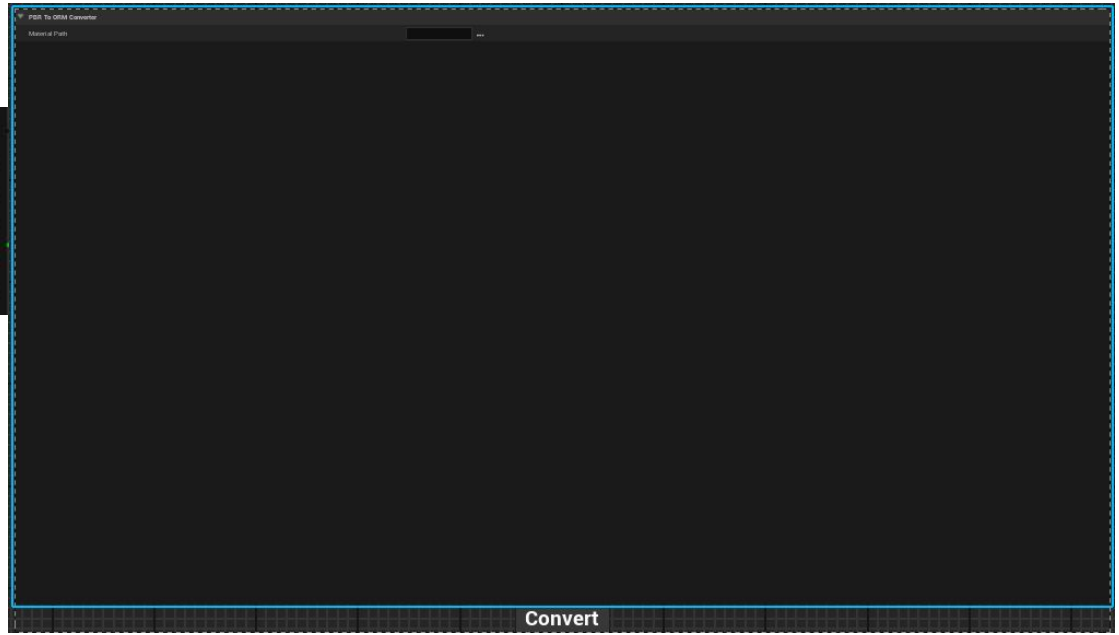
プロジェクトの方針決定：
- タスクの理解
    - PBRテクスチャとORMテクスチャの仕組みに関する調査
- 解決策のリサーチ
    1. Editor Utility Widget (Blueprint) + C++
    2. Editor Utility Widget (Blueprint) + Python　（経験無）
    3. Editor Utility Widget (Blueprint) + Easy Texture Packer　（有料）

１番目の選択肢に決定

2日目

Editor Utility Widgetのデザイン作成





２日目

ディレクトリへのシンプルなアクセスロジック

２日目

C++での関数作成
- エラー発生
    - C++の関数を作成する際の問題点
    - ビルドエラーが発生し、プロジェクトが開けなくなってしまいました。
- 解決法
    - 不足していた依存関係をインストール
    - プロジェクトのディレクトリを変更。日本語の文字が含まれるとビルドができなるなる

2日目～4日目

開発とデバグ
以下の関数をBlueprintCallableとして作成：
- ConvertPBRtoORM
- GetAssetPathFromFolder
- ReplacePBRwithORMinMaterialInstance

```
UFUNCTION(BlueprintCallable, meta = (DisplayName = "Convert PBR to ORM"))

static UTexture2D* ConvertPBRToORM(FString AOFilePath, FString RoughnessFilePath, FString MetallicFilePath, const FString& SavePath);

UFUNCTION(BlueprintCallable, meta = (DisplayName = "Replace PBR With ORM In MaterialInstance"))

static void ReplacePBRWithORMInMaterialInstance(UMaterialInstanceConstant* MaterialInstance, UTexture2D* NewORMTexture);

UFUNCTION(BlueprintCallable, meta = (DisplayName = "Get Asset Path From Folder"))
1 Blueprint reference
static UMaterialInstanceConstant* GetAssetPathFromFolder(FString FolderPath);
```

ConvertPBRtoORM

プロジェクトの主要な関数。Roughness、AO、Metalicのテクスチャマップのファイルパスから、新しいORMテクスチャマップを作成

```
UTexture2D* UPBRToORM::ConvertPBRToORM(FString AOFilePath, FString RoughnessFilePath, FString MetallicFilePath, const FString& SavePath)
{
#if WITH_EDITOR
    FString AOFileRelativePath = GetRelativePath(AOFilePath);
    FString RoughnessFileRelativePath = GetRelativePath(RoughnessFilePath);
    FString MetalicFileRelativePath = GetRelativePath(MetallicFilePath);
    UTexture2D* AO = Cast<UTexture2D>(StaticLoadObject(UTexture2D::StaticClass(), nullptr, *AOFileRelativePath));
    UTexture2D* Roughness = Cast<UTexture2D>(StaticLoadObject(UTexture2D::StaticClass(), nullptr, *Ro
    UTexture2D* Metallic = Cast<UTexture2D>(StaticLoadObject(UTexture2D::StaticClass(), nullptr, *Met

    AO->CompressionSettings = TextureCompressionSettings::TC_VectorDisplacementmap;
    AO->MipGenSettings = TextureMipGenSettings::TMGS_NoMipmaps;
    AO->SRGB = false;
    AO->UpdateResource();

    Roughness->CompressionSettings = TextureCompressionSettings::TC_VectorDisplacementmap;
    Roughness->MipGenSettings = TextureMipGenSettings::TMGS_NoMipmaps;
    Roughness->SRGB = false;
    Roughness->UpdateResource();

    Metallic->CompressionSettings = TextureCompressionSettings::TC_VectorDisplacementmap;
    Metallic->MipGenSettings = TextureMipGenSettings::TMGS_NoMipmaps;
    Metallic->SRGB = false;
    Metallic->UpdateResource();
```

```
    if (!AO || !Roughness || !Metallic) return nullptr;

    const int32 Width = AO->GetPlatformData()->SizeX;
    const int32 Height = AO->GetPlatformData()->SizeX;

    UE_LOG(LogTemp, Error, TEXT("Surface Size - Width: %d, Height: %d"), Width, Height);

    if (Roughness->GetSizeX() != Width || Metallic->GetSizeX() != Width)
    {
        UE_LOG(LogTemp, Error, TEXT("All textures must be same size."));
        return nullptr;
    }
    const int32 TotalPixels = Width * Height;
    const uint8* RoughData = Roughness->Source.LockMip(0);
    const uint8* MetalData = Metallic->Source.LockMip(0);
    const uint8* AOData = AO->Source.LockMip(0);
    if (!RoughData || !MetalData || !AOData)
    {
        UE_LOG(LogTemp, Error,
            TEXT("Failed to access source data of one or more textures."));
        Roughness->Source.UnlockMip(0);
        Metallic->Source.UnlockMip(0);
        AO->Source.UnlockMip(0);
        return nullptr;
    }
```

ConvertPBRtoORM

```cpp
// Determine source pixel format for each (to handle bytes per pixel)
ETextureSourceFormat roughFmt = Roughness->Source.GetFormat();
ETextureSourceFormat metalFmt = Metallic->Source.GetFormat();
ETextureSourceFormat aoFmt = AO->Source.GetFormat();
// Prepare buffer for combined texture pixels (4 bytes per pixel, BGRA8 format)
TArray<uint8> CombinedPixels;
CombinedPixels.AddUninitialized(TotalPixels * 4);
// Lambda to fetch a single grayscale value from source data (supports G8 or BGRA8 formats)
auto GetGray = [&](const uint8* Data, ETextureSourceFormat Format, int32 index)->uint8
    {
        switch (Format)
        {
        case TSF_G8: // 8-bit grayscale
            return Data[index];
        case TSF_G16: // 16-bit grayscale (take high byte as a
            return Data[index * 2];
        case TSF_BGRA8: // 8-bit BGRA (use R channel)
        case TSF_BGRE8:
            return Data[index * 4 + 2]; // R channel of BGRA
        default:
            // For other formats, just return first byte (migh
            return Data[index];
        }
    };
```

```cpp
for (int32 i = 0; i < TotalPixels; ++i)
{
    uint8 roughVal = GetGray(RoughData, roughFmt, i);
    uint8 metalVal = GetGray(MetalData, metalFmt, i);
    uint8 aoVal = GetGray(AOData, aoFmt, i);
    CombinedPixels[i * 4 + 2] = roughVal; // Red channel
    CombinedPixels[i * 4 + 1] = metalVal; // Green channel
    CombinedPixels[i * 4 + 0] = aoVal; // Blue channel
    CombinedPixels[i * 4 + 3] = 0xFF; // Alpha channel (opaque)
}
UE_LOG(LogTemp, Error, TEXT("Pixeling"));

// Unlock source data arrays
Roughness->Source.UnlockMip(0);
Metallic->Source.UnlockMip(0);
AO->Source.UnlockMip(0);

UE_LOG(LogTemp, Error, TEXT("Creating the ORM texture"));
FString FolderPath = FPackageName::GetLongPackagePath(AOFileRelativePath);
FString FolderName = FPaths::GetCleanFilename(FolderPath);

FString NewTexName = FolderName + TEXT("_ORM");
FString NewTexPackagePath = FolderPath / NewTexName;
UPackage* NewTexPackage = CreatePackage(*NewTexPackagePath);
NewTexPackage->FullyLoad();
UTexture2D* NewORMTexture = NewObject<UTexture2D>(NewTexPackage, *NewTexName, RF_Public | RF_Standalone | RF_MarkAsRootSet);
```

ConvertPBRtoORM

```
// Initialize texture properties and allocate MIP data
NewORMTexture->AddToRoot(); // prevent garbage collection



FTexturePlatformData* TexturePlatformData = new FTexturePlatformData();
TexturePlatformData->SizeX = Width;
TexturePlatformData->SizeY = Height;
//TexturePlatformData->NumSlices = 1;
TexturePlatformData->PixelFormat = PF_B8G8R8A8;
 // Allocate first mipmap
 FTexture2DMipMap* Mip = new FTexture2DMipMap();
 Mip->SizeX = Width;
 Mip->SizeY = Height;
 Mip->BulkData.Lock(LOCK_READ_WRITE);
 void* NewTexData = Mip->BulkData.Realloc(TotalPixels * 4);
 FMemory::Memcpy(NewTexData, CombinedPixels.GetData(), TotalPixels * 4);
 Mip->BulkData.Unlock();
 TexturePlatformData->Mips.Add(Mip);

 NewORMTexture->SetPlatformData(TexturePlatformData);

 UE_LOG(LogTemp, Error, TEXT("Successfully set platformdata"));

 NewORMTexture->SRGB = false;
 NewORMTexture->CompressionSettings = TC_Masks;
 NewORMTexture->MipGenSettings = TMGS_NoMipmaps;
 // Initialize source art data for the texture asset (store uncompressed source pixels)
 NewORMTexture->Source.Init(Width, Height, /*NumSlices=*/1, /*NumMips=*/1,
     TSF_BGRA8, CombinedPixels.GetData());
 NewORMTexture->UpdateResource();

 UE_LOG(LogTemp, Error, TEXT("Created textures"));
 // Save the new texture asset to Content Browser
 FAssetRegistryModule::AssetCreated(NewORMTexture);
 NewTexPackage->MarkPackageDirty();
 FString FilePath =
     FPackageName::LongPackageNameToFilename(NewTexPackagePath,
```

```
FSavePackageArgs SaveArgs;
SaveArgs.TopLevelFlags = RF_Public | RF_Standalone;
SaveArgs.Error = GError;
SaveArgs.bWarnOfLongFilename = true;
SaveArgs.SaveFlags = SAVE_NoError;
UPackage::SavePackage(NewTexPackage, NewORMTexture, *FilePath, SaveArgs); // save .uasset to disk
UE_LOG(LogTemp, Error, TEXT("Saved  the texture"));
```

GetAssetPathFromFolder

補助関数です。マテリアルインスタンスを取得するために使用されます。

```
UMaterialInstanceConstant* UPBRToORM::GetAssetPathFromFolder(FString FolderPath)
{
    // Caminho completo da pasta "Content"
    FString ContentDir = FPaths::ConvertRelativePathToFull(FPaths::ProjectContentDir());

    // Validação: a pasta precisa estar dentro de /Content
    if (!FolderPath.StartsWith(ContentDir))
    {
        UE_LOG(LogTemp, Error, TEXT("Path is not inside /Content folder."));
        return nullptr;
    }

    // Remove o prefixo absoluto e obtém caminho relativo a /Game
    FString RelativePath = FolderPath;
    RelativePath.RemoveFromStart(ContentDir);
    RelativePath = RelativePath.Replace(TEXT("\\"), TEXT("/"));

    // Remove barra final, se tiver
    RelativePath.RemoveFromEnd(TEXT("/"));

    // Pega o nome da última pasta
    FString FolderName = FPaths::GetCleanFilename(RelativePath);

    // Monta o caminho no formato /Game/Path/To/Folder/FolderName.FolderName
    FString ObjectPath = FString::Printf(TEXT("/Game/%s/%s.%s"), *RelativePath, *FolderName, *FolderName);

    UMaterialInstanceConstant* MaterialInstance = Cast<UMaterialInstanceConstant>(StaticLoadObject(UMaterialInstanceConstant::StaticClass(), nullptr, *ObjectPath));

    return MaterialInstance;
}
```

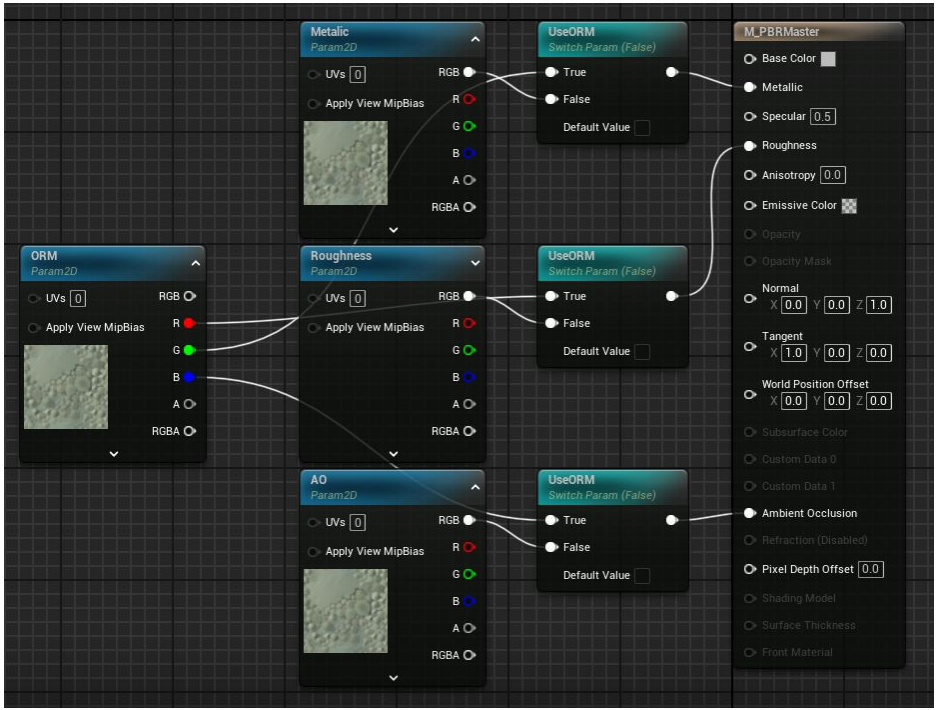
ReplacePBRwithORMinMaterialInstance

PBRテクスチャを新しいORMテクスチャに置き換え。

```cpp
void UPBRToORM::ReplacePBRWithORMInMaterialInstance(UMaterialInstanceConstant * MaterialInstance, UTexture2D* NewORMTexture)
{
#if WITH_EDITOR
    if (!MaterialInstance || !NewORMTexture)
    {
        UE_LOG(LogTemp, Error, TEXT("Invalid input to ReplacePBRWithORMInMaterialInstance."));
        return;
    }

    // Define os nomes dos parâmetros PBR
    static const FName RoughnessParamName("Roughness");
    static const FName MetallicParamName("Metalic");
    static const FName AOParamName("AO");

    //// Limpa apenas os parâmetros PBR se estiverem definidos
    //MaterialInstance->ClearParameterValueEditorOnly(FMaterialParameterInfo("Roughness"));
    //MaterialInstance->ClearParameterValueEditorOnly(FMaterialParameterInfo("Metallic"));
    //MaterialInstance->ClearParameterValueEditorOnly(FMaterialParameterInfo("AO"));

    TArray<FName> PBRParams = { FName("Metalic"), FName("Roughness"), FName("AO") };

    MaterialInstance->TextureParameterValues.RemoveAll([&](const FTextureParameterValue& Param) {
        return PBRParams.Contains(Param.ParameterInfo.Name);
        });

    SetUseORM(MaterialInstance, true);

    // Define o novo parâmetro ORM
    static const FName ORMParamName("ORM");
    MaterialInstance->SetTextureParameterValueEditorOnly(ORMParamName, NewORMTexture);

    // Atualiza o material na interface
    MaterialInstance->PostEditChange();
    MaterialInstance->MarkPackageDirty();
#endif
}
```

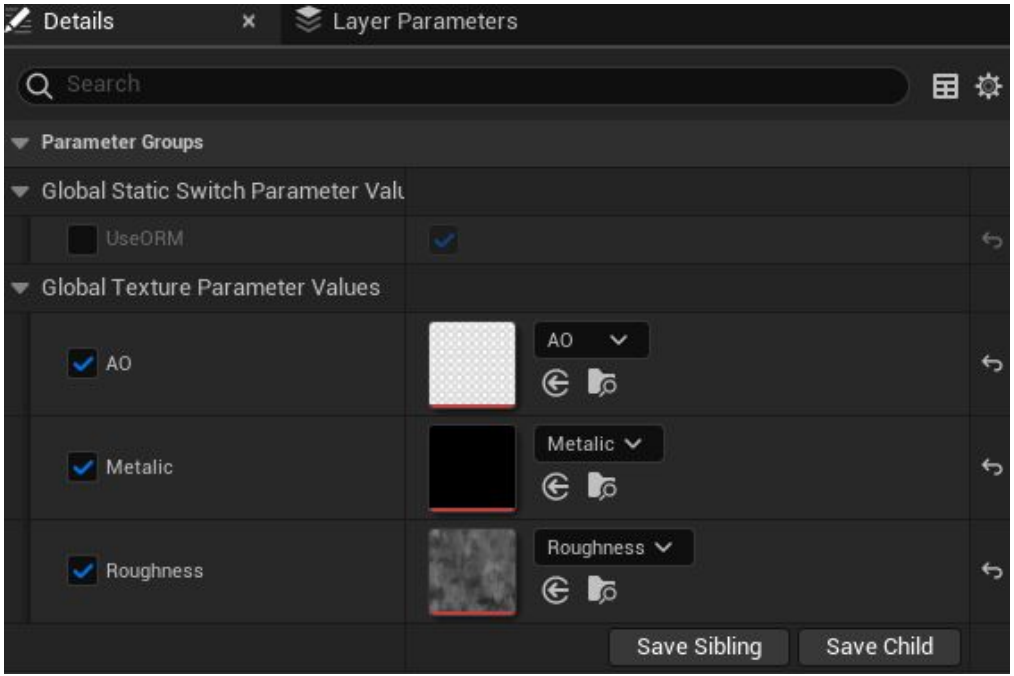
テスト

マテリアルインスタンスの元になるマテリアルベース



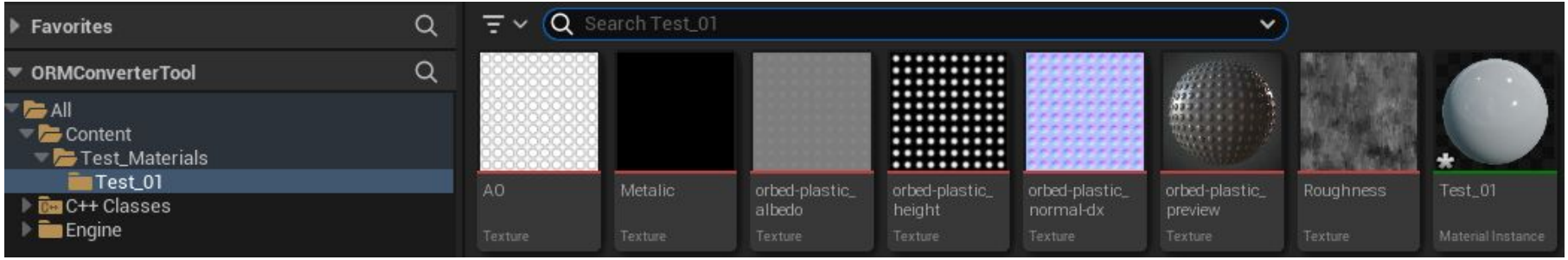
テスト

マテリアルインスタンス

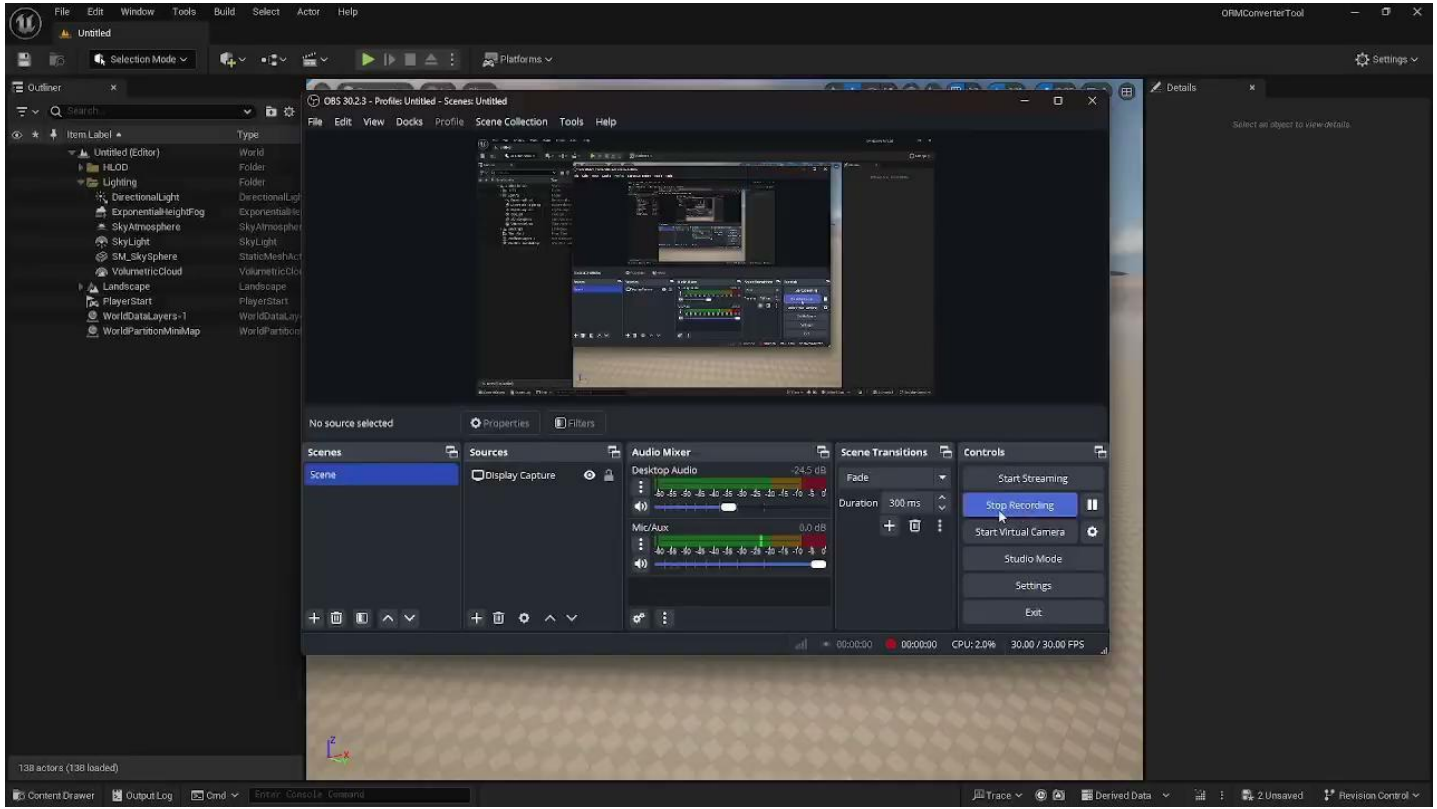


テスト

テクスチャー


Favorites
ORMConverterTool
All
Content
Test_Materials
Test_01
C++ Classes
Engine
Search Test_01
AO
Texture
Metalic
Texture
orbed-plastic_albedo
Texture
orbed-plastic_height
Texture
orbed-plastic_normal-dx
Texture
orbed-plastic_preview
Texture
Roughness
Texture
Test_01
Material Instance

テスト


File Edit Window Tools Build Select Actor Help
Untitled
ORMConverterTool
Selection Mode
Platforms
Settings
Outliner
Item Label
Type
Untitled (Editor)
World
HLOD
Folder
Lighting
Folder
DirectionalLight
ExponentialHeightFog
SkyAtmosphere
SkyLight
SM_SkySphere
VolumetricCloud
Landscape
PlayerStart
WorldDataLayers-1
WorldPartitionMiniMap
OBS 30.2.3 - Profile: Untitled - Scenes: Untitled
File Edit View Docks Profile Scene Collection Tools Help
No source selected
Properties
Filters
Scenes
Scene
Sources
Display Capture
Audio Mixer
Desktop Audio
-24.5 dB
Mic/Aux
0.0 dB
Scene Transitions
Fade
Duration
300 ms
Controls
Start Streaming
Stop Recording
Start Virtual Camera
Studio Mode
Settings
Exit
00:00:00
00:00:00
CPU: 2.0%
30.00 / 30.00 FPS
Details
Select an object to view details.
138 actors (138 loaded)
Content Drawer
Output Log
Cmd
Enter Console Command
Trace
Derived Data
2 Unsaved
Revision Control

困ったこと

パソコンはあまり性能が良くなく、動作が遅い。

採用されたら、新しいパソコンを買うつもりです。