



**UNREAL**  
ENGINE

# Unreal Engine 5.2

## 注目機能の紹介

- GTMF 2023 -

エピック ゲームズ ジャパン  
岡田 和也, 鈴木 孝司

# はじめに

本講演では UE5.2 における機能追加・改善の  
すべてを紹介することは **しません**

UE5.2 リリースノートから（個人的に）重要なものを紹介したり  
少し分かりづらい項目の補足などを行います

The image shows the cover of the Unreal Engine 5.2 Release Notes document. It features a dark, atmospheric cityscape at night with illuminated buildings and a hazy sky. The text is overlaid in white and light blue.

## Unreal Engine 5.2 リリースノート

Unreal Engine 5.2 の新機能および更新機能の概要

<https://docs.unrealengine.com/5.2/ja/unreal-engine-5.2-release-notes/>

# Nanite, Lumen などのレンダリング機能や PCG はこの後の講演で

## Unreal Engine 5.2 アップデート ～Rendering/PCG～

エピック ゲームズ ジャパン

プランナー

プログラマー(クライアント)

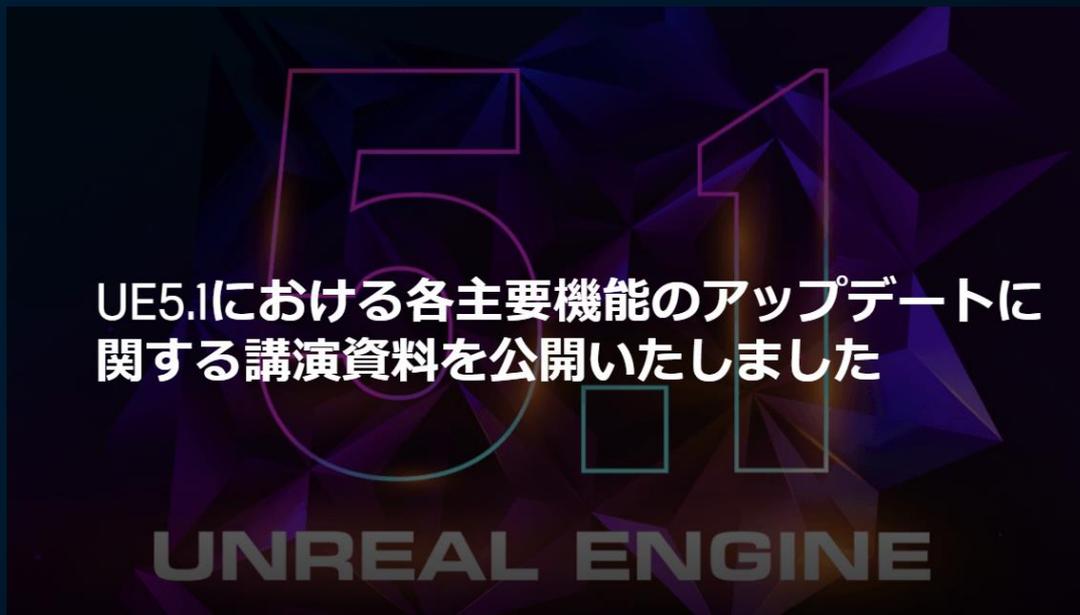
デザイナー(2D/3D)

Unreal Engine 5.2では様々な新機能・改善が入り、グラフィックス関連ではLumenやNaniteといったUE5の注目機能が更に強化され、新たにSubstrateと呼ばれるマテリアルの作成手法が導入されました。

その他にも大規模なワールドを効率的に作成するためのPCG(Procedural Content Generation Framework)など様々な機能が追加されており、本講演ではそれらの中から特にオススメしたい機能についてご紹介いたします。

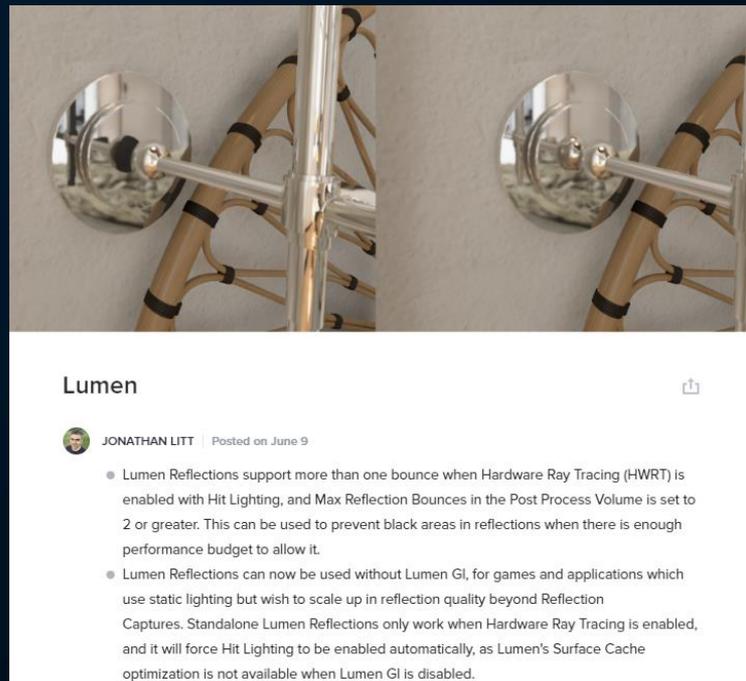
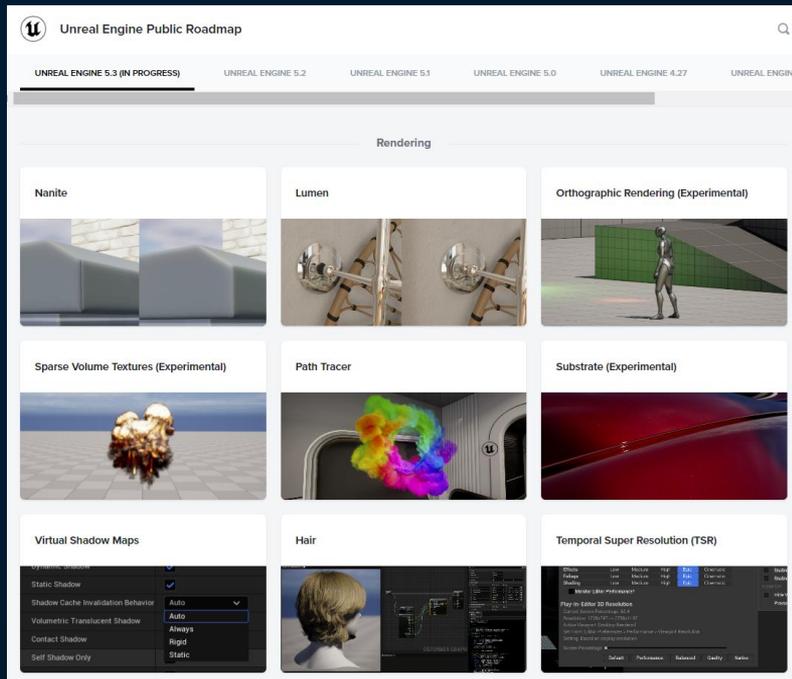


# UE5.1 におけるアップデート情報はこちら



<https://www.unrealengine.com/ja/events/epicgamesjapan-ue5-1-updateinfo>

# UE5.3 におけるアップデート情報はこちら



<https://portal.productboard.com/epicgames/1-unreal-engine-public-roadmap/tabs/88-unreal-engine-5-3-in-progress>



**UNREAL**  
ENGINE

# Animation

UE5.2 アップデート

# Python リターゲティングのサポート

# Python リターゲティングのサポート

カスタム仕様の Python スクリプトを実装して、IK リグを使ったリターゲティング プロセスを自動化および高速化できるようになりました



# Python リターゲティングのサポート

カスタム仕様の Python スクリプトを実装して、IK リグを使ったリターゲティング プロセスを自動化および高速化できるようになりました

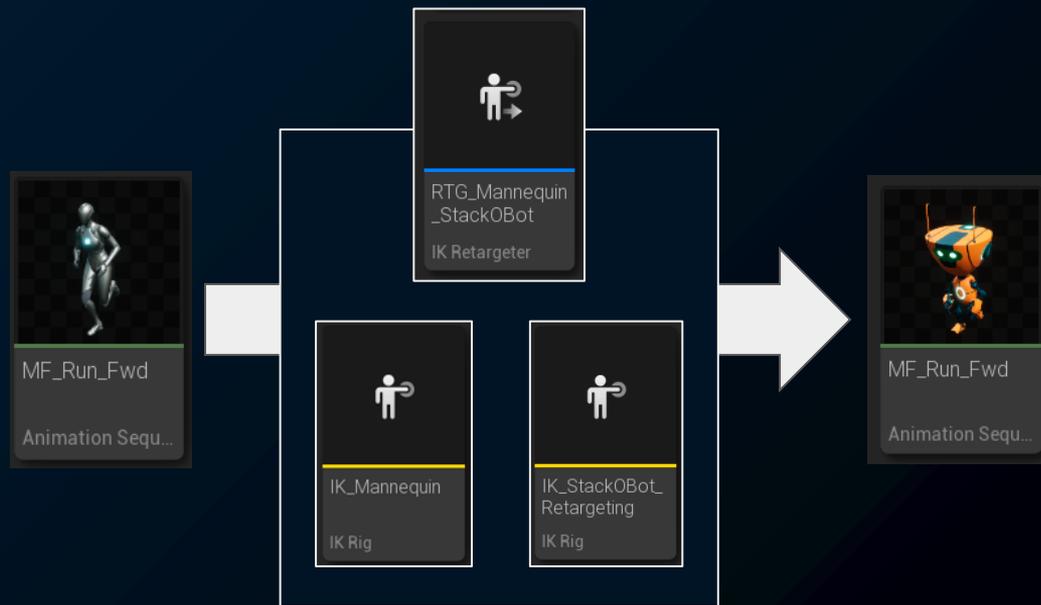


IK Rig ・ IK Retargeter を Blueprint または Python から自動生成したりリターゲット処理を実行できるようになりました



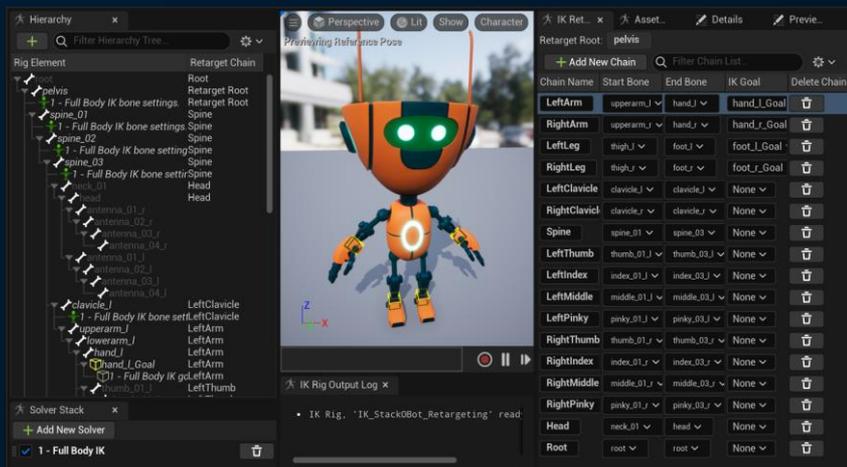
# IK Rig / IK Retargeter

ボーン構造が異なっても、アニメーションを流用するための仕組み  
UE5でのアニメーションの流用方法について



# IK Rig / IK Retargeter のこれまでの課題

- セットアップ項目が多い
- Skeleton毎にアセットを作成・調整する必要がある



IK Rig



IK Retargeter

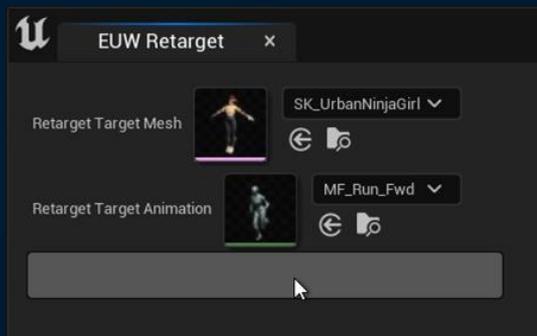
# UE5.2から BP / Python で生成・設定可能に



```
fbik_index = ikr_controller.add_solver(  
    unreal.IKRigFBIKSolver)  
ikr_controller.set_root_bone("pelvis", 0)
```

```
ikr_controller.add_new_goal("hand_l_goal",  
    "hand_l")  
ikr_controller.add_new_goal("hand_r_goal",  
    "hand_r")
```

# ボタン1つで、面倒な作業を自動化・実行！



EditorUtilityで  
作成した自作ツール



設定済みのIK Rig, Retargeterを自動生成  
アニメーションリターゲットを実行



<https://www.youtube.com/watch?v=It-j0OICUf4>

# 学習リソース

## Python で IK リグを使用する

<https://docs.unrealengine.com/5.2/ja/using-python-to-create-and-edit-ik-rigs-in-unreal-engine/>

## IKリターゲッターで Python を使用する

<https://docs.unrealengine.com/5.2/ja/using-python-to-create-and-edit-ik-retargeter-assets-in-unreal-engine/>

## BPで自動生成・実行するサンプル

<https://dev.epicgames.com/community/snippets/J55V/unreal-engine-generate-ikrig-and-ikretargeter-with-blueprint-bp-ikrig-ikretargeter>

## 新しいIKR アセットを作成する

新しいIK リグ アセットを作成するには、次のファクトリを使用できます。

```
# アセット ツールを取得します。

asset_tools = unreal.AssetToolsHelpers.get_asset_tools()

# ファイル パスで定義された場所に IK リグを作成します。例: `.../Ga

ikr = asset_tools.create_asset(asset_name='IK_Mannequin'

package_path='/Game/', asset_class=unreal.IKRigDefinition

factory=unreal.IKRigDefinitionFactory())
```

# UE5.2の新機能を活かした生成ツールが！



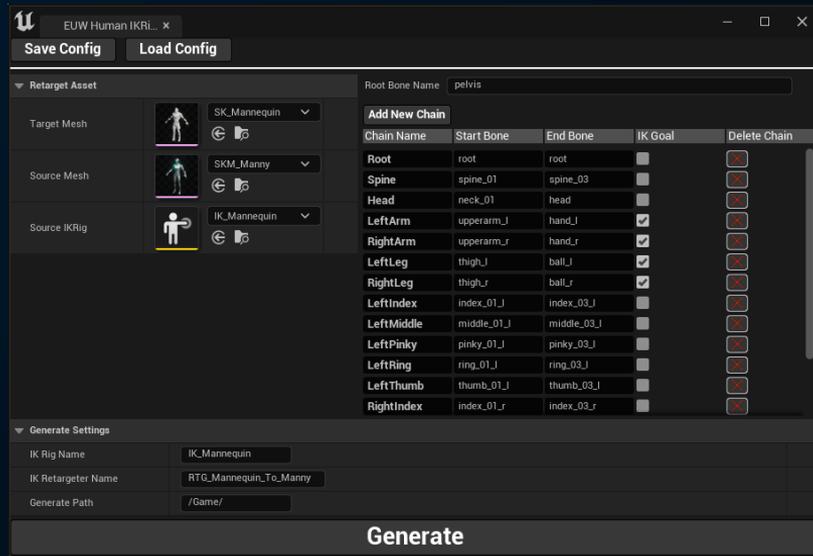
Leon Gameworks

@Leon\_Gameworks

IKRig,IKRetargeterを生成できるツールをGithubに公開しました！  
チェーンの追加や設定のセーブ、ロードなどが可能です。  
[github.com/LeonGameworks/...](https://github.com/LeonGameworks/)

Epic Games Japanのおかずさんが公開された情報を元に作成しています！  
貴重な情報を公開して頂きありがとうございます！  
[dev.epicgames.com/community/snip...](https://dev.epicgames.com/community/snip...)  
#UE5

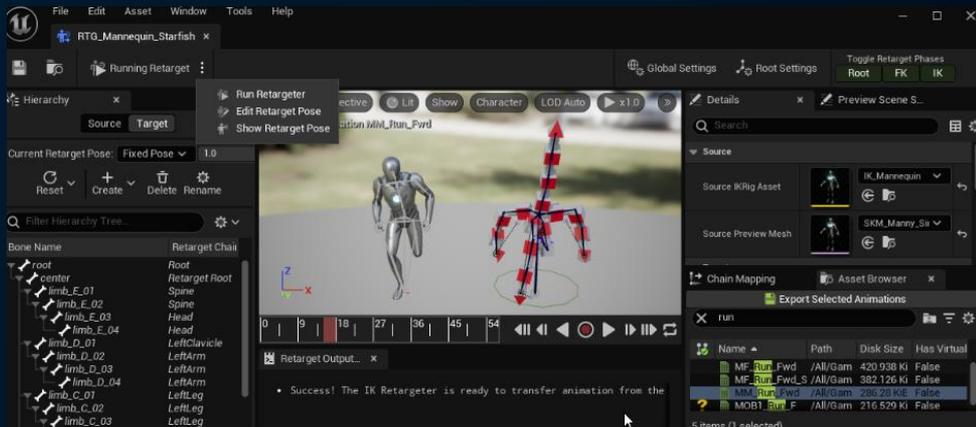
[UE専門のゲームデベロッパーである株式会社Leon Gameworks様のツイート](https://github.com/LeonGameworks/Human_IKRigGenerator)



[https://github.com/LeonGameworks/Human\\_IKRigGenerator](https://github.com/LeonGameworks/Human_IKRigGenerator)

UE5.3にて  
IK Retargeterエディタへの  
機能追加など  
様々な改善が入る予定

Unreal Engine Public Roadmap より  
<https://portal.productboard.com/epicgames/1-unreal-engine-public-roadmap/c/1076-animation-retargeting>



## Animation Retargeting



GREG RICHARDSON | Posted on April 18

This release focuses on quality-of-life improvements for Retargeting.

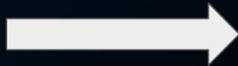
- Toolbar in Retarget Editor - Easy access to Global and Root Settings along with the Retarget Phases. Mode switches between Running the Retarget, editing the Retarget Pose, and viewing the Retarget Pose.
- Retarget Poses - Edit a bone's local space to zero out bone rotations.
- Create IK Rig/IK Retargeter assets from a selected Skeletal Mesh/IK Rig in the Right Mouse Context Menu.
- Thumbnails for IK Rig and IK Retargeter assets.

# スケルトンの互換性の改善 ( Compatible Skeleton )

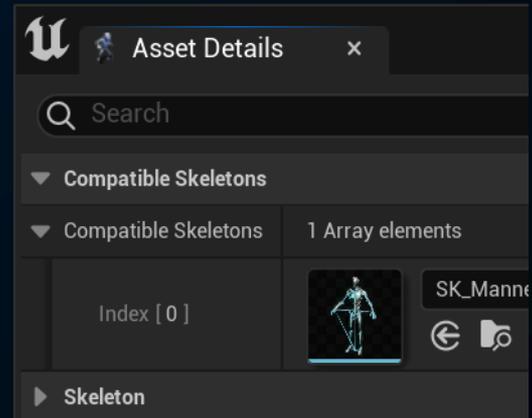
# スケルトンの互換性について

Skeletonアセットが異なっても

**Compatible (互換性) があるから アニメーションの共有はOK!** という機能



UNREAL ENGINE



設定画面

<https://www.docswell.com/s/EpicGamesJapan/KQN3EK-UE5-ShareAnimation>

# スケルトンの互換性に関する改善点

## 互換性の設定が双方向に

- Skeleton A → B の設定をすれば、A ← B も互換性アリ状態に

## 互換性の設定をしなくても、 他Skeletonのアニメーションが利用可能に

- Skeleton が異なっても  
ボーン名・構造の比較でアニメーション再生をするように

## 互換性の設定がEditor Onlyに

- リマッピング処理がランタイム上で実行されるように
- 余計な参照関係を回避

# 学習リソース



@EGJ-Kaz\_Okada (Kaz Okada)

posted at 2023-04-07 updated at 2023-04-10

**[UE5] UE5.2で更に便利になった 互換スケルトン (Compatible Skeletons)について**

UnrealEngine, ue5

[https://qiita.com/EGJ-Kaz\\_Okada/items/387b73923089c4622352](https://qiita.com/EGJ-Kaz_Okada/items/387b73923089c4622352)

Beta

# 機械学習 (ML) デフォーマ

# MLデフォーマ

事前に機械学習したデータを用いて  
高品質のメッシュ変形を**模倣**する機能

UE5.1でβ版をリリース。**5.2で改良**



# MLデフォーマ サンプルを公開



## ML デフォーマ サンプル

Epic Games - UE 機能サンプル - 2023/05/11

機械学習デフォーマ (Machine Learning Deformer) サンプルは、Unreal Engine で最新の ML テクノロジーを使用する方法を紹介します。

サポートされたプラットフォーム



サポートされたエンジンバージョン

5.2

ランチャーで開く

<https://www.unrealengine.com/marketplace/ja/product/ml-deformer-sample>

# 学習リソース

## 公式ドキュメント「機械学習デフォーマの使い方」

<https://docs.unrealengine.com/5.2/ja/how-to-use-the-machine-learning-deformer-in-unreal-engine/>

## Nearest Neighbor Model (NNM)モデルの解説

<https://dev.epicgames.com/community/learning/tutorials/2lJy/unreal-engine-nearest-neighbor-model>

## MetaHuman Framework & Machine Learning for Next-Gen Character Deformation | GDC 2023

<https://www.youtube.com/watch?v=OmMi6E0EkQw>

# UE5.3 で 様々な最適化・ 改善を追加する予定です

Unreal Engine Public Roadmap より  
<https://portal.productboard.com/epicgames/1-unreal-engine-public-roadmap/c/1162-ml-deformer>

## ML Deformer



TB

TIM BRAKENSIEK | Posted on June 9

ML Deformer enables users to approximate a complex rig, nonlinear deformer, or any arbitrary deformation by training a Machine Learning model that runs in real time in UE.

Improvements in UE 5.3 include:

- The addition of a masking system for Local Neural Morph Model, which improves the ability of users to work with structured data
- Improvements to the Nearest Neighbor model UI and workflow
- The removal of dependency on NNI framework
- The ability to create training masks, and visualize the masks per bone / bone group / curve in the viewport
- The ability to evaluate optimized trained neural morph models on all platforms that support morph targets (PC, Xbox, PS5, Android, iOS, Switch, etc.)
- Training masks operate as expected: a limited mask for a bone limits deformation to the masked region for the given bone
- The ability to create and train new models using the Nearest Neighbor and Neural Morph model on Windows
- The ability to open the ML Deformer Sample Showcase and retrain all shipped models without error on Windows
- There is no longer any need to take any manual steps (for instance, installing a Python package) for the Nearest Neighbor and Neural Morph models to train correctly on Windows
- The ability to run trained Neural Morph and Nearest Neighbor models on Linux and macOS (although training errors may be encountered)

# MetaHuman Animator を公開

iPhone 1台で忠実度の高いパフォーマンス キャプチャを実現

<https://www.unrealengine.com/ja/blog/delivering-high-quality-facial-animation-in-minutes-metahuman-animator-is-now-available>





# Introducing MetaHuman Animator



<https://www.youtube.com/watch?v=WghYEFloaM8>

# 学習リソース

## How to Use MetaHuman Animator in Unreal Engine

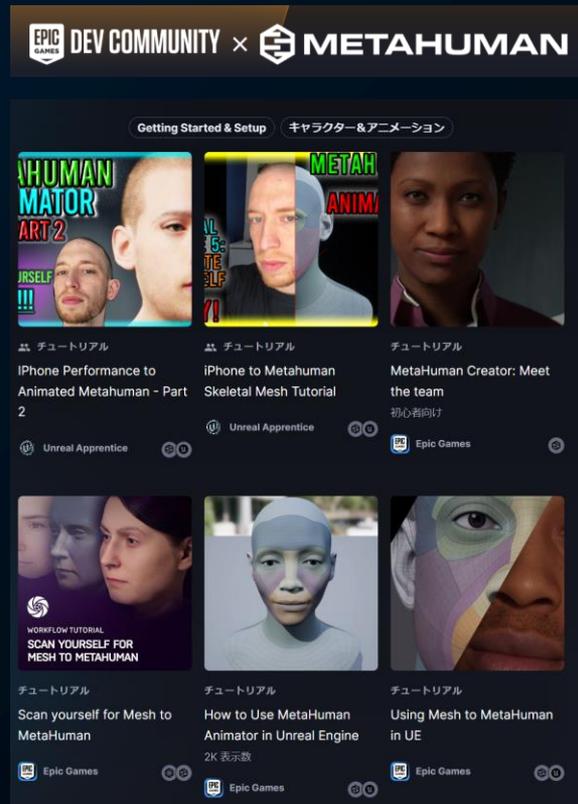
<https://www.youtube.com/watch?v=WWLF-a68-CE>

## Hellblade actor demos MetaHuman Animator | State of Unreal | GDC 2023

<https://www.youtube.com/watch?v=uAYI2rOtgm4>

## MetaHuman ハブ

<https://dev.epicgames.com/community/metahuman>





**UNREAL**  
ENGINE

# Modeling / Editor

UE5.2 アップデート

Beta

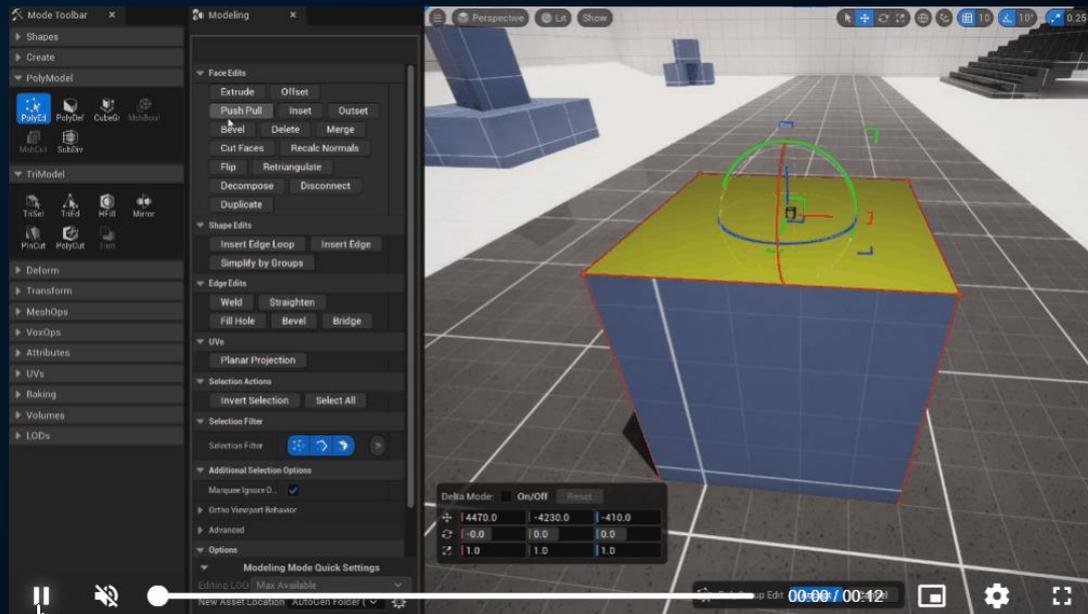
# モデリングモード Update

# モデリングモードとは

エディタ上で  
モデル作成・編集・最適化が  
できる専用のモード・機能

- メッシュ生成・編集・変形
- 自動UV展開、UV編集
- ピボット（基準位置）編集
- バイク処理
- リダクション
- テクスチャへのバイク
- LOD生成・調整

など

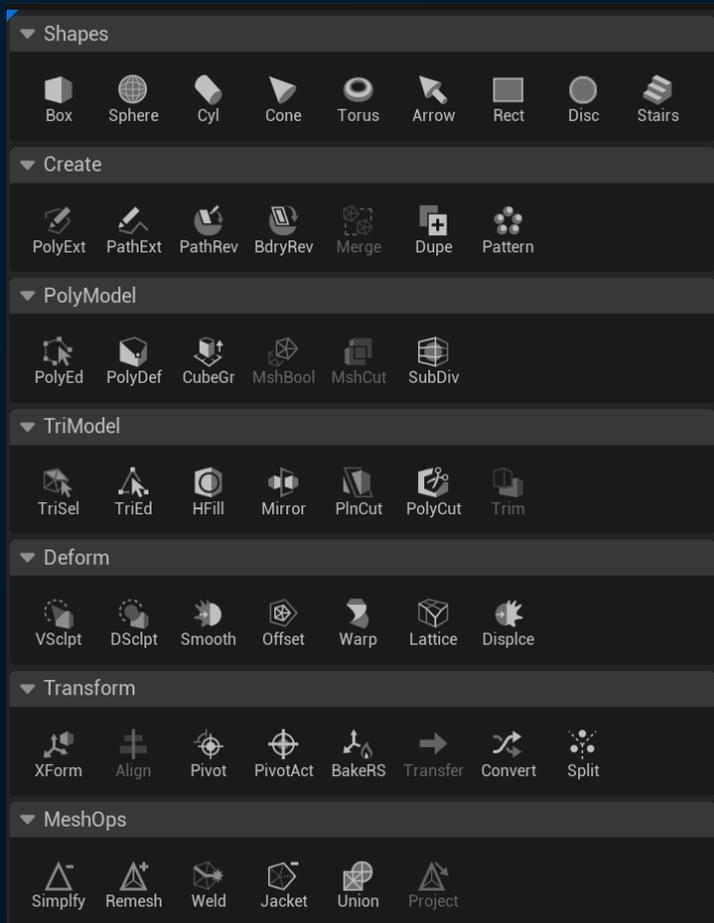


# 80種類の ツールを提供！

<https://docs.unrealengine.com/5.2/ja/modeling-tools-in-unreal-engine/>

ツールパレットが  
多くなりすぎたので

UE5.3で  
UIが一新予定



モデリングモードが目指すもの

レベルデザイン が  
そのレベル内 で 完結すること

# BSP から モデリングモード へ

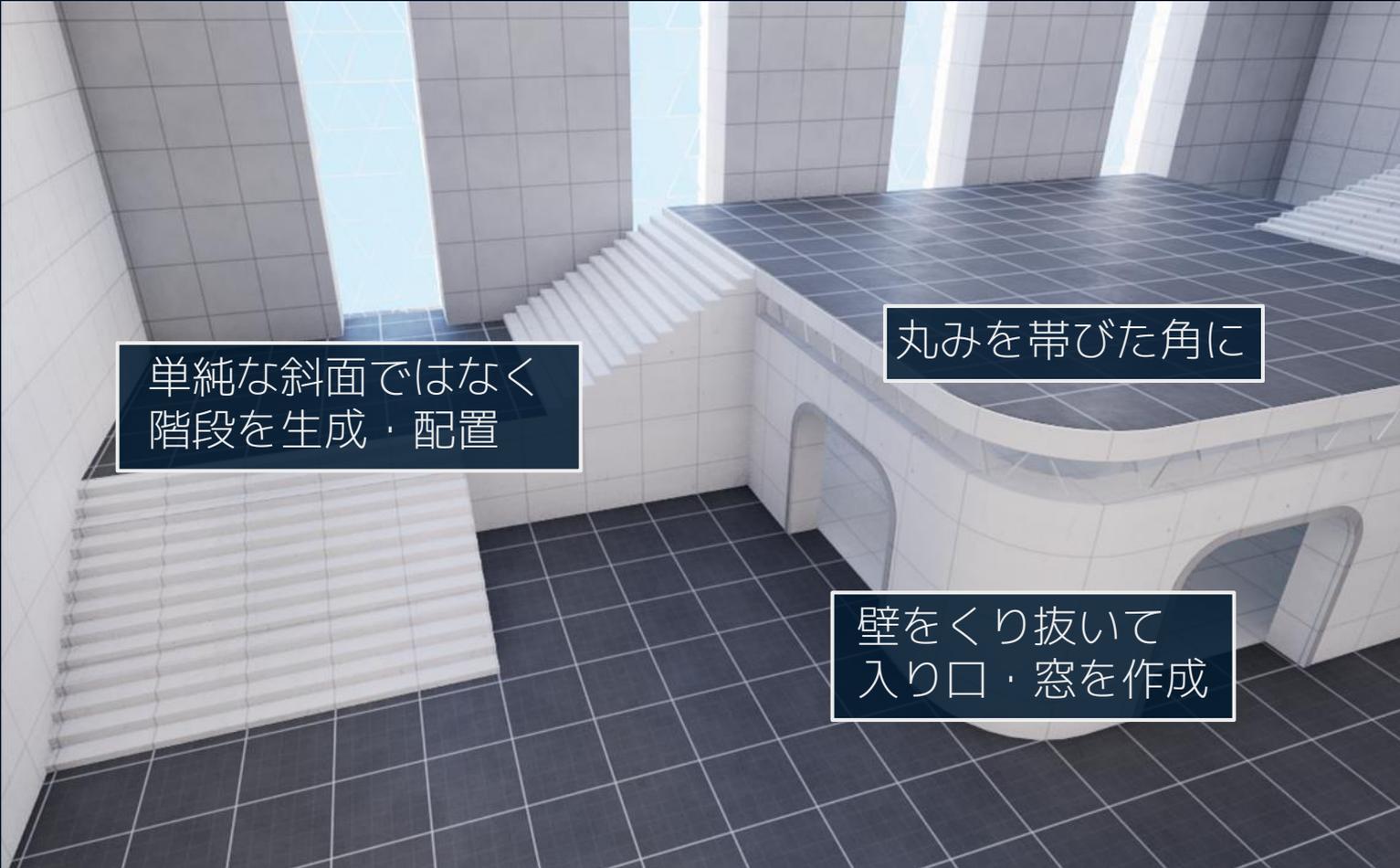
近年のレベルデザイン・ジオメトリの複雑さに対応するには  
より高性能・多機能なツールが必要に



Unreal Tournament 4(2014)



Lyra Starter Game (2022)



単純な斜面ではなく  
階段を生成・配置

丸みを帯びた角に

壁をくり抜いて  
入り口・窓を作成

# ハイポリメッシュを扱う難しさ



Megascansで  
キットバッシングしたレベル

この辺りのメッシュを  
少し曲げたいなあ…



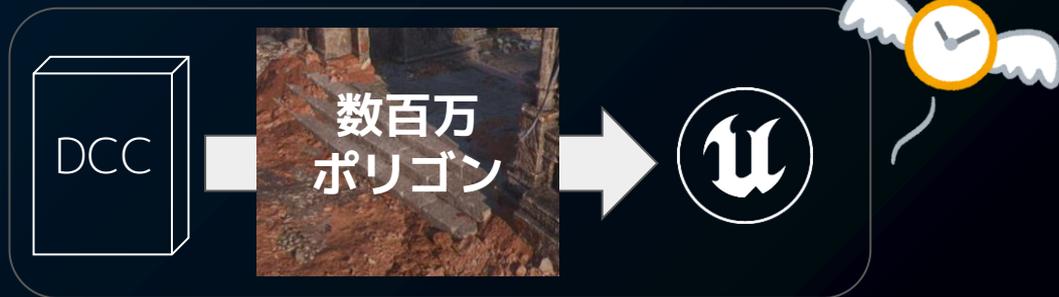
ハイポリメッシュの  
エクスポート(UE)  
インポート(DCC)



ハイポリメッシュと  
相性が悪い  
環境・ツールでの作業



ハイポリメッシュの  
エクスポート(DCC)  
インポート(UE)



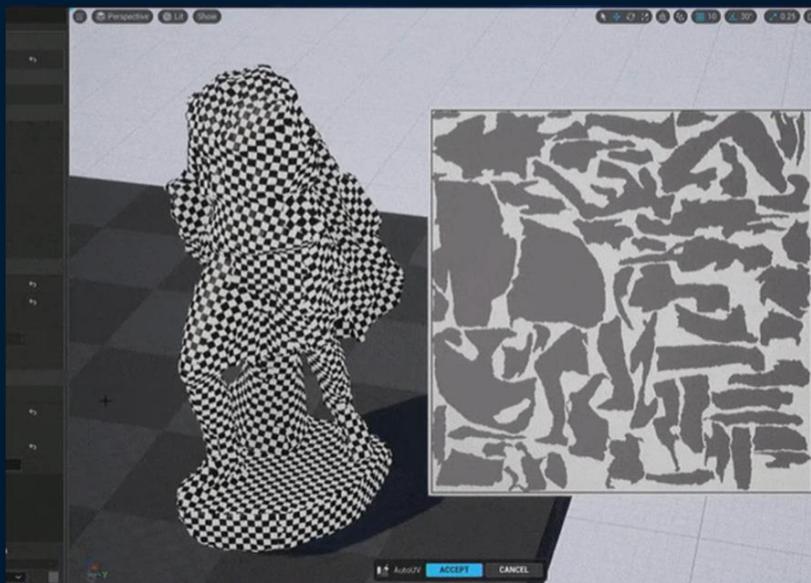
# DCCツールを介さないことで、作業を高速化

## UE上で作業を完結させるメリット

- ツール間を行き来する際の待ち時間を排除
- ハイポリメッシュをリアルタイムで編集可能
- 編集後の結果を最終クオリティですぐ確認可能



ハイポリメッシュでも  
高速なイテレーションを実現！



ハイポリメッシュを自動UV展開



スキャンデータをクリーンアップ  
ハイポリメッシュをリダクション

# モデリングモードが実現すること

- レベルデザイン用のモデルを生成・編集
- インポートしたアセットのカスタマイズ
- DDCツールを介さない高速な反復作業
- ハイポリメッシュの最適化



# モデリングモードが **実現しない** こと

従来のDCCツールを使った  
モデリング・ワークフローの置き換えは「しない」

プロジェクト固有のヒーローキャラクターや  
ランドマークとなるモデルを作るツールではない



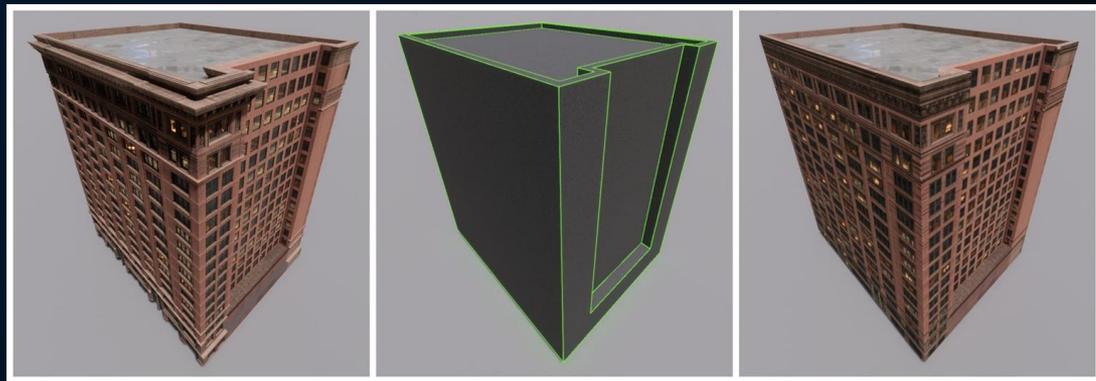
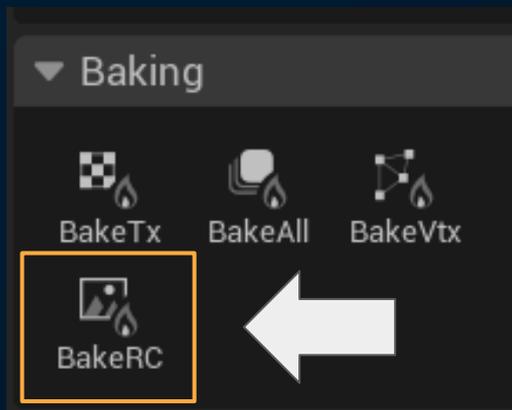
Beta

# モデリングモード Update

# BakeRC (レンダリング キャプチャ バイク ツール)

[UE5.2リリースノート](#)より

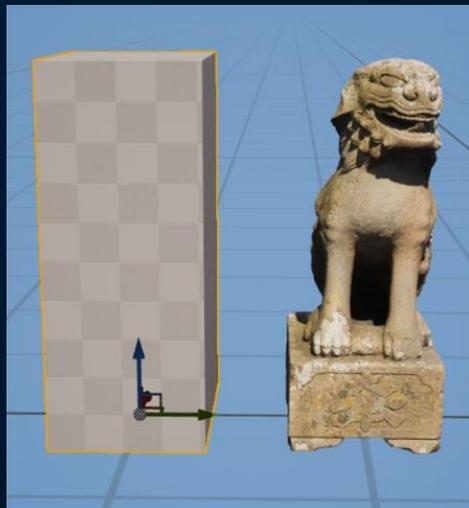
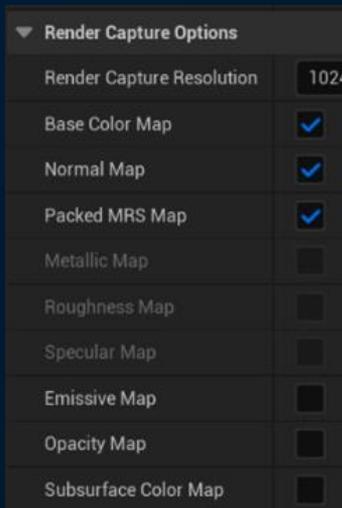
BakeRC では、バーチャル フォトやレンダリング キャプチャを介して、複数のソース メッシュからターゲット メッシュのテクスチャをバイクします



# BakeRC の仕組み

## 複数視点からの描画結果をテクスチャにバイク

BaseColorだけでなく、NormalやMetallicなどもバイク可能



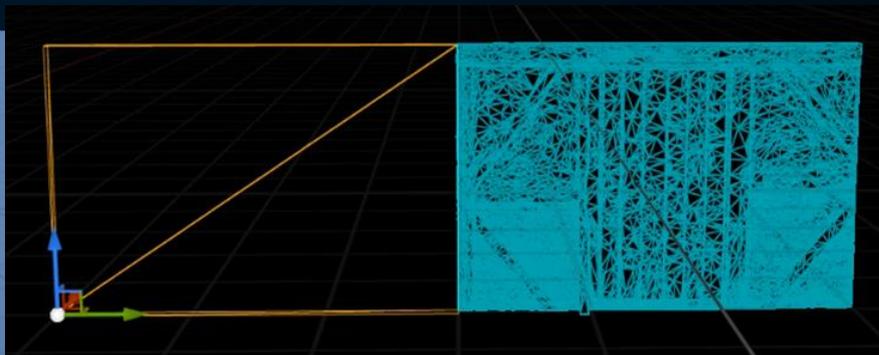
# BakeRC の メリット ・ 使い方

ベイクしたテクスチャを使うことで

**ローポリ**メッシュでハイポリメッシュの見た目を（ある程度）再現できる

使い方：

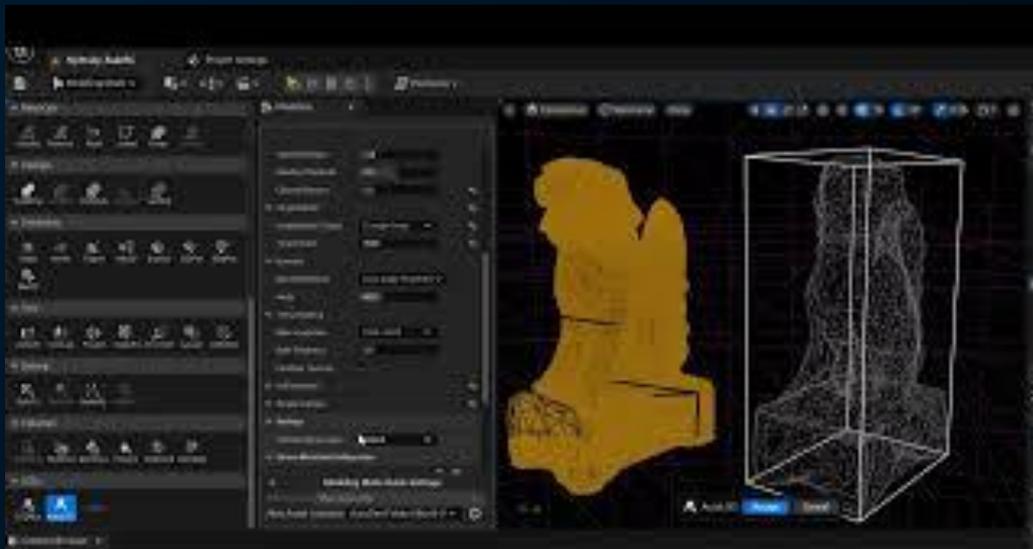
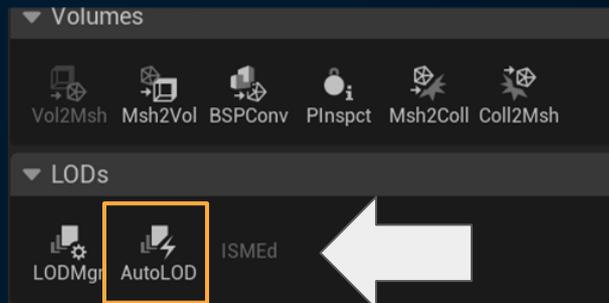
<https://twitter.com/pafuhana1213/status/1671717634503737344>



# Auto LOD

ハイポリメッシュから  
ローポリ版を作成する機能

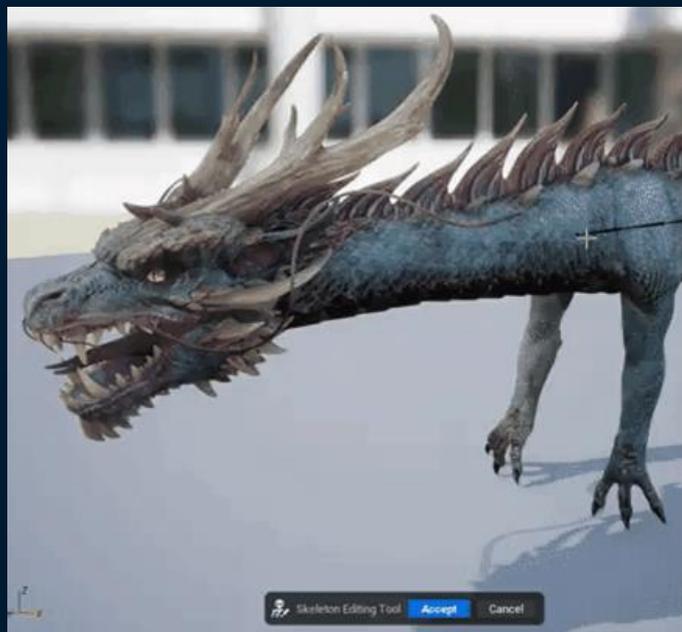
- 豊富なリダクション設定
  - テクスチャも対象
- 自動UV展開
- 法線のテクスチャベイク
- リアルタイムで設定反映



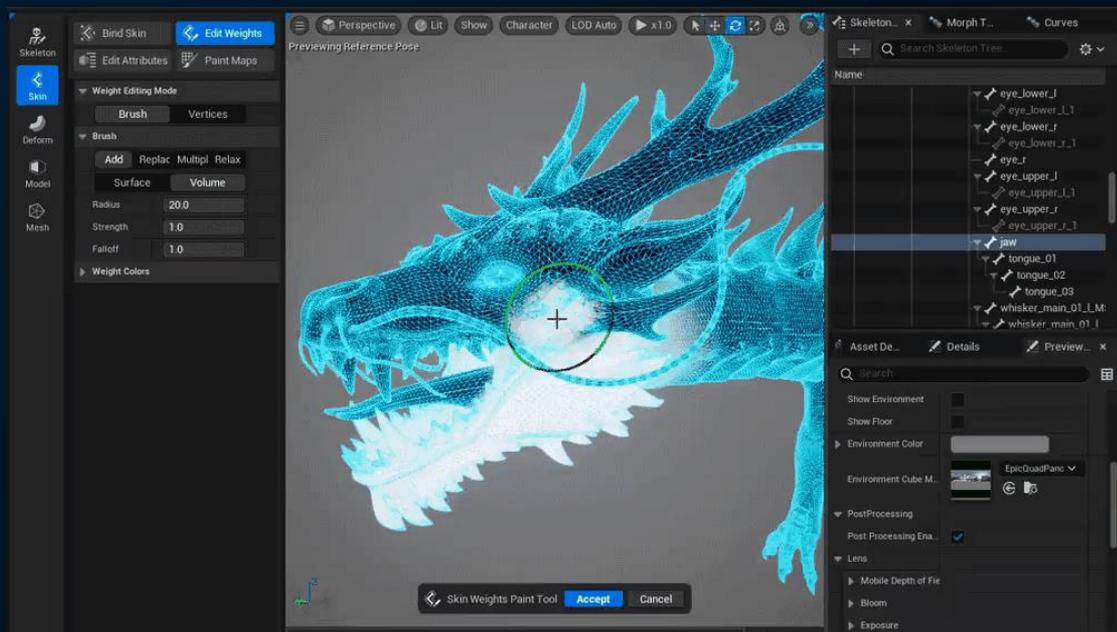
<https://www.youtube.com/watch?v=xiVJMLIMBI>

<https://twitter.com/pafuhana1213/status/1670484073410879488>

# UE5.3 では Skeletal Editor を追加



スケルトンの作成・編集



スキンウェイトの作成・編集

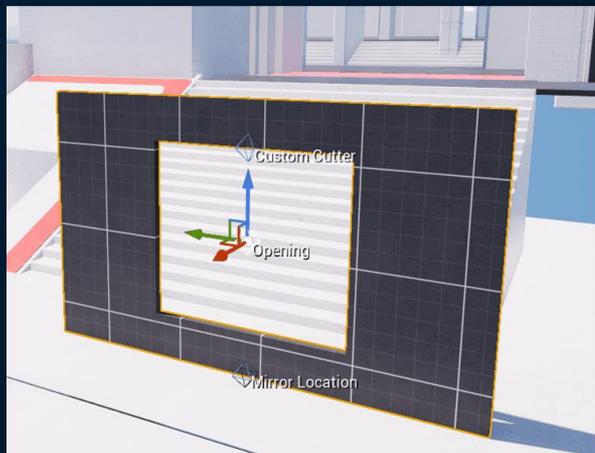
Experimental

# Geometry Script Update

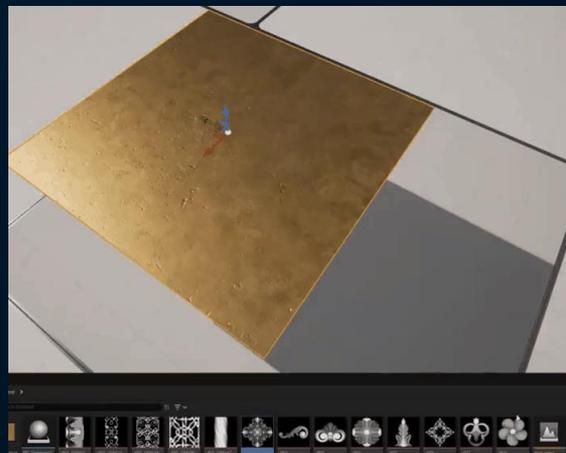
# Geometry Scriptとは

Blueprint / Python でメッシュを動的に生成・編集できる機能

レベルデザインだけでなく、プロシージャルモデリング用途でも利用可能



Lyra のジオメトリ ツール



Alpha / Height mapから  
メッシュ作成



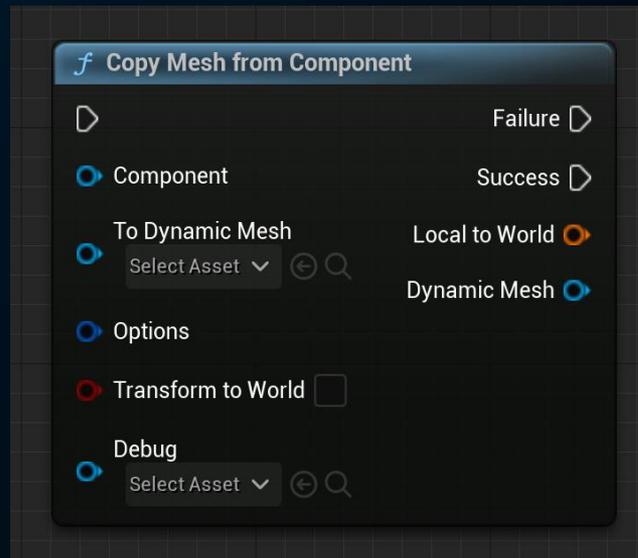
草木 ジェネレータ

# Copy Mesh From Component の改善

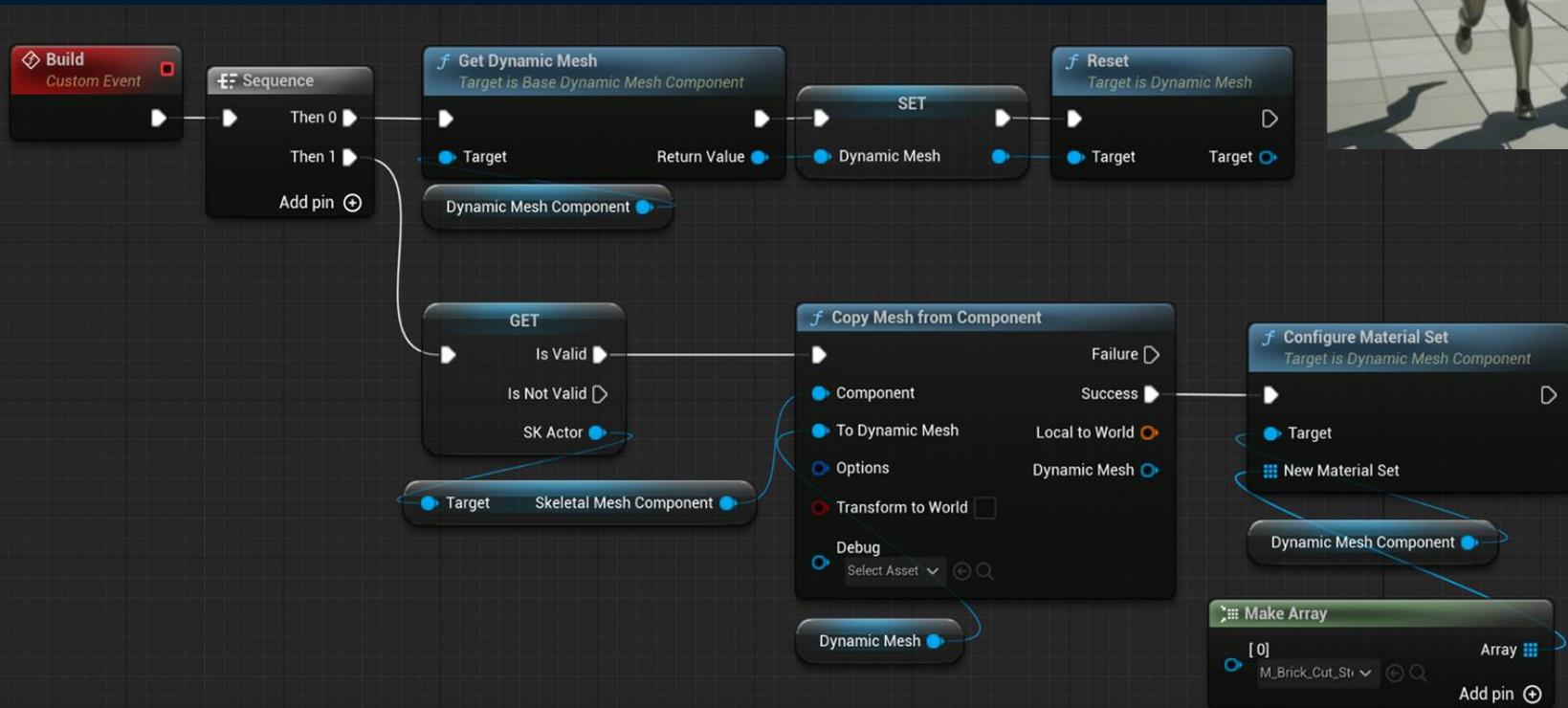
指定のComponent から メッシュ情報を取得し  
Geometry Script が扱う Dynamic Mesh にコピーするノード

## 対応しているComponent :

- Static Mesh
- Dynamic Mesh
- Brush
- **Spline Mesh** **New !**
- **Posed Skinned Mesh** **New !**
- **Skeletal Mesh** **New !**



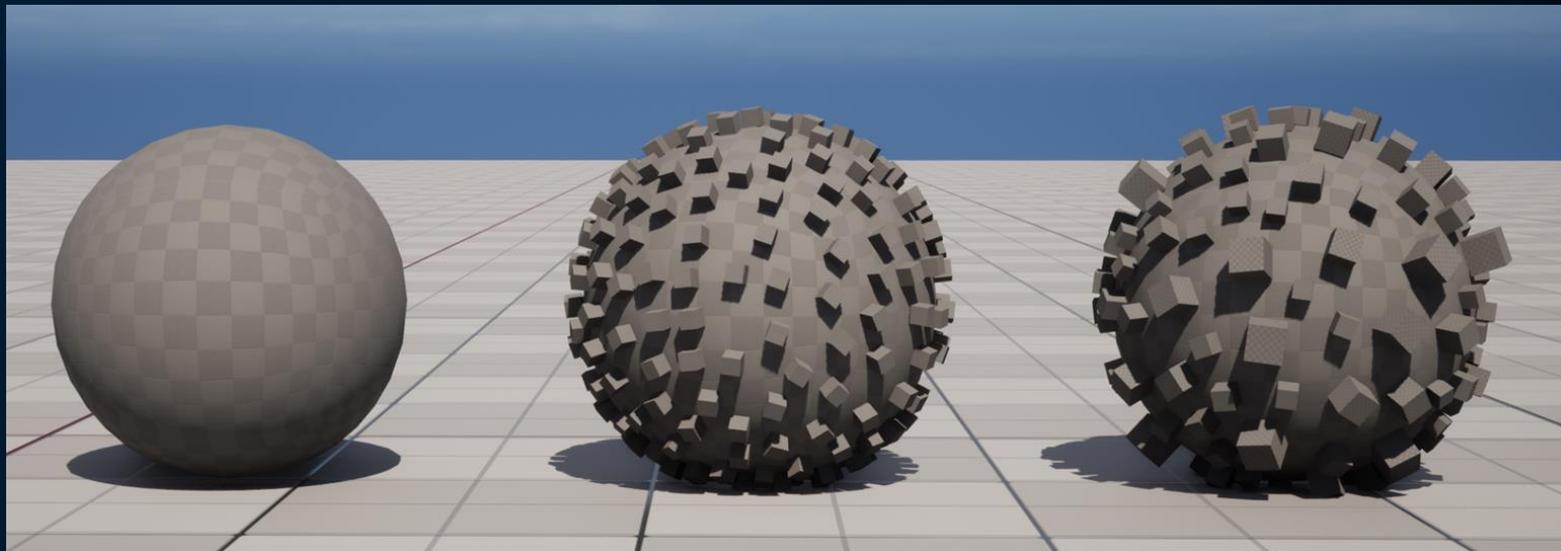
# Skeletal Mesh Component から ポーズ付きメッシュ情報をコピー



# メッシュ表面のサンプリング

任意の 3D メッシュ上で

「均等 or 不均等」 かつ 「交差せず」 にサンプル点を生成する機能



Compute Point Sampling :

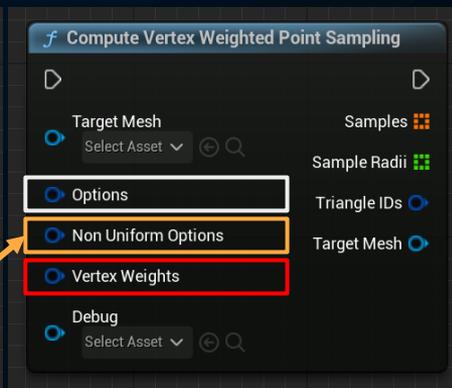
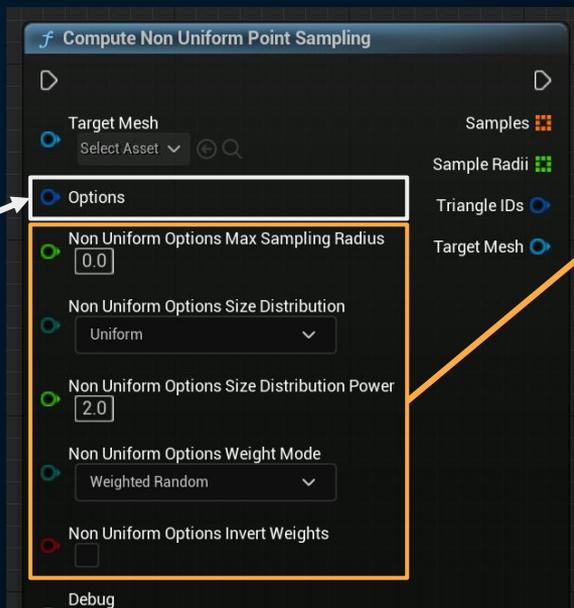
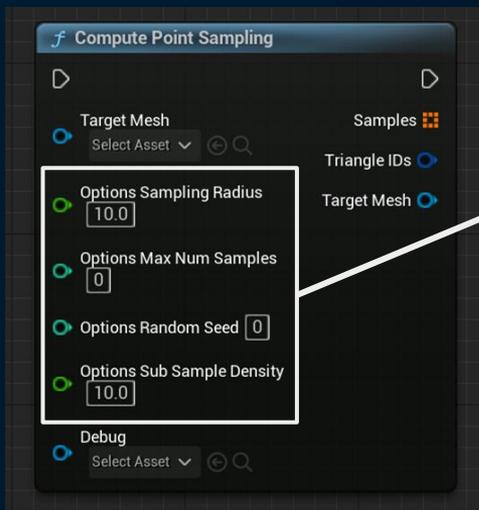
**均等**に分布

Compute **Non Uniform** Point Sampling :

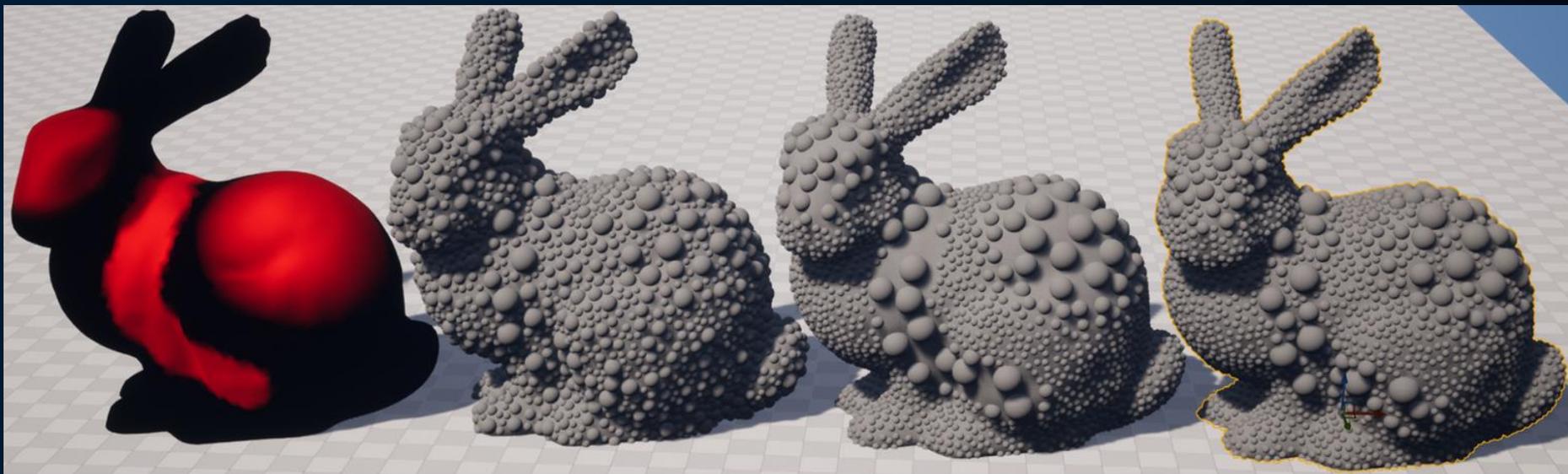
**不均等**に分布

Compute **Vertex Weighted** Point Sampling :

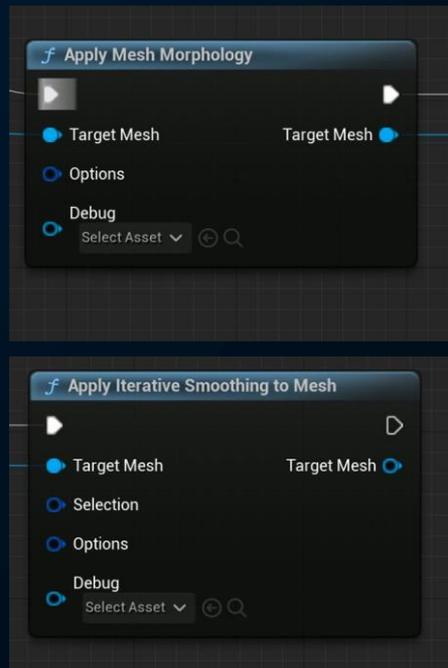
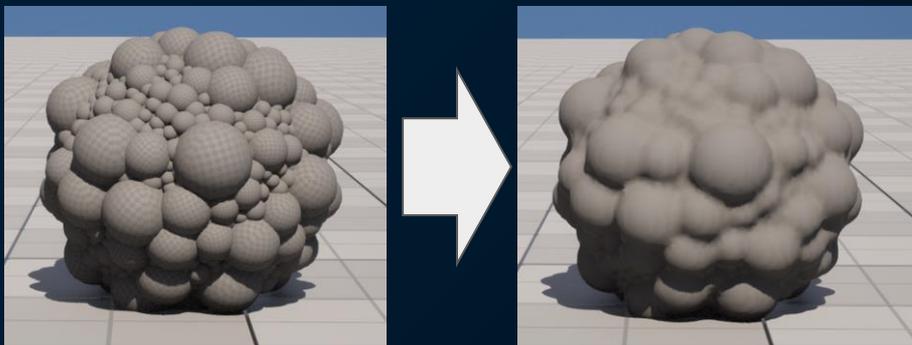
**不均等** かつ **頂点毎に重み付け** して分布



# 頂点カラーの R値 を 頂点ごとの重み付けに利用した例



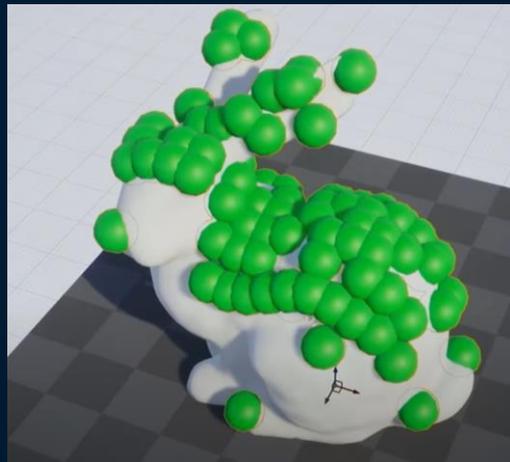
# サンプリング結果を元に配置したメッシュに対して SDFベースの結合処理 や スムージングを行うと…



SDFベースの  
結合処理

スムージング処理  
を複数回実行

# 活用例



[Modeling and Geometry Scripting in UE:  
Past, Present, and Future](#)

[Geometry Script を使って  
岩の上に雪を配置するデモ・解説](#)

# ジオメトリ スクリプトのリファレンス

Geometry Script は  
現時点で **250 近くのノードを提供**

基本的なものは  
公式ドキュメントに解説あり

<https://docs.unrealengine.com/5.2/ja/geometry-script-reference-in-unreal-engine/>

## コンテンツ

アセットおよびコンポーネントの読み取り/書き込み

プリミティブ生成

トランスフォームおよび変形

コンポジション/分解

メッシュ モデリング

細分化関数

単純化関数

法線関数

クリーンアップ/修復関数

低レベルメッシュクエリ

低レベルメッシュの構築

MaterialID 関数

頂点カラー

UV 関数

PolyGroup 関数

メッシュのジオメトリクエリ

メッシュの比較

BVH/AABBTree および空間クエリ

ユーティリティ

# 学習リソース

## 公式ドキュメント「モデリングとジオメトリ スクリプト」

<https://docs.unrealengine.com/5.2/ja/modeling-and-geometry-scripting-in-unreal-engine/>

## Exploring Geometry Tools in UE5

<https://www.youtube.com/watch?v=apCSgAAkDTU>

## Introduction to Geometry Scripting

<https://www.youtube.com/watch?v=gDso9X4HgLA>

## Modeling and Geometry Scripting in UE: Past, Present, and Future

<https://www.youtube.com/watch?v=pEC3krOwcbM>

## Blockout and Asset Production in UE5

<https://www.youtube.com/watch?v=R5TsbW4fk8>

## 開発者の Ryan さんの Youtubeチャンネル

<https://www.youtube.com/@RyanSchmidtEpic>

## UE5攻略リンク

<https://ue5study.com/unrealengine-modeling/>

<https://ue5study.com/unrealengine-geometry-script/>

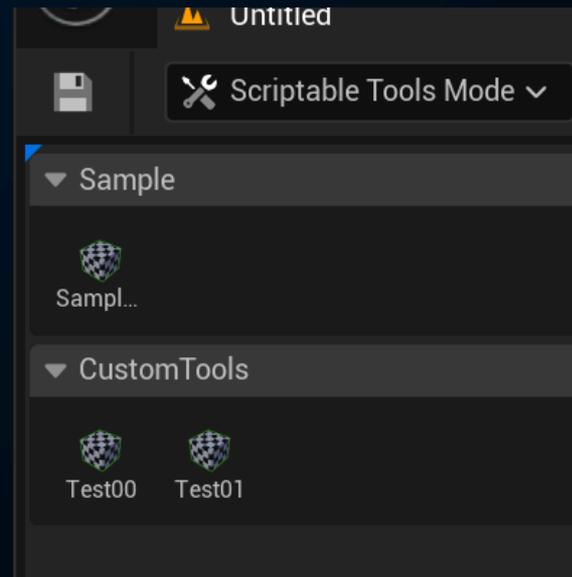
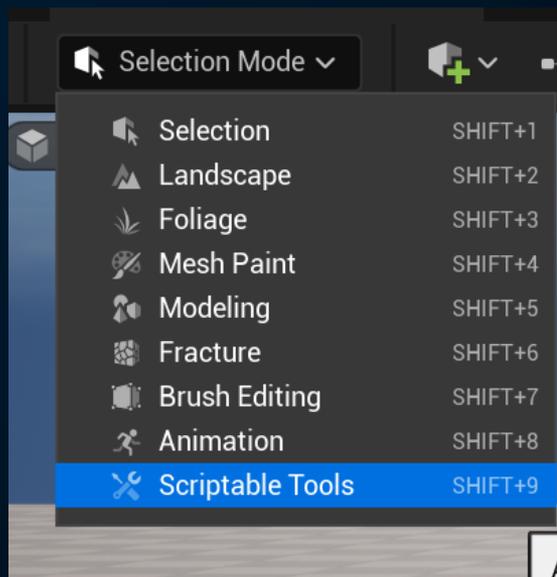
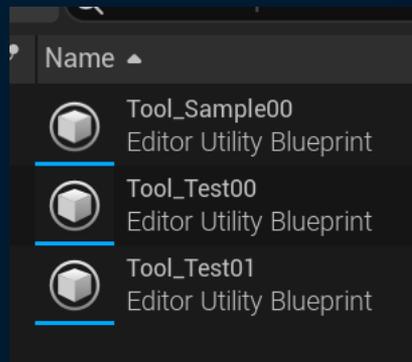
Experimental

# スクリプト可能ツールの フレームワーク (Scriptable Tools Framework)

# Scriptable Tools Framework

BP / C++ を使って

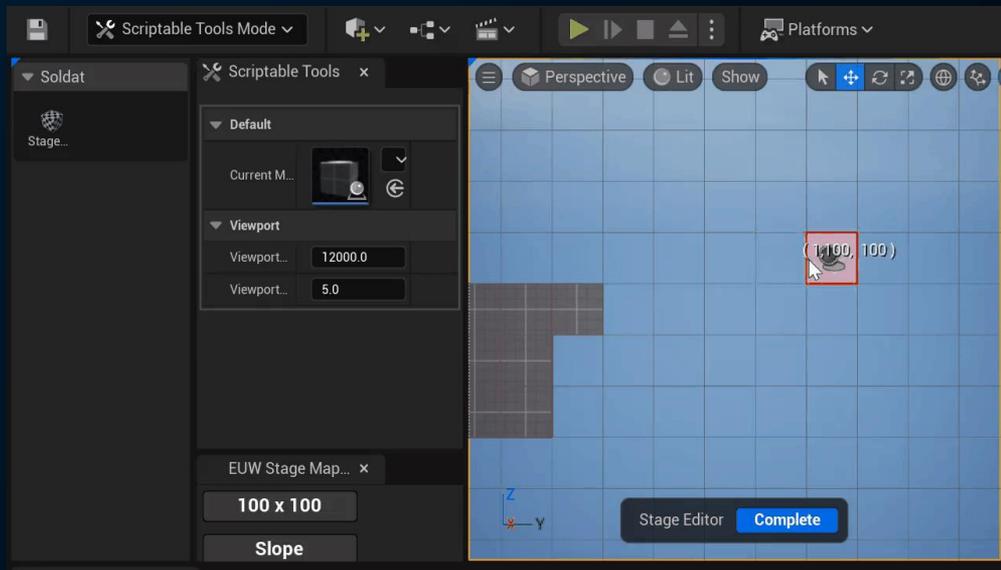
**プロジェクト独自の エディタモード・ツール** を作成可能に！



# 活用例



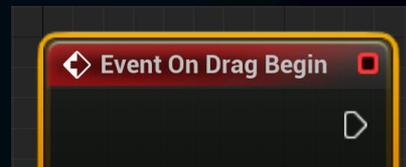
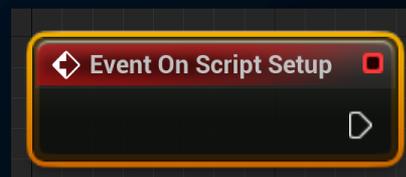
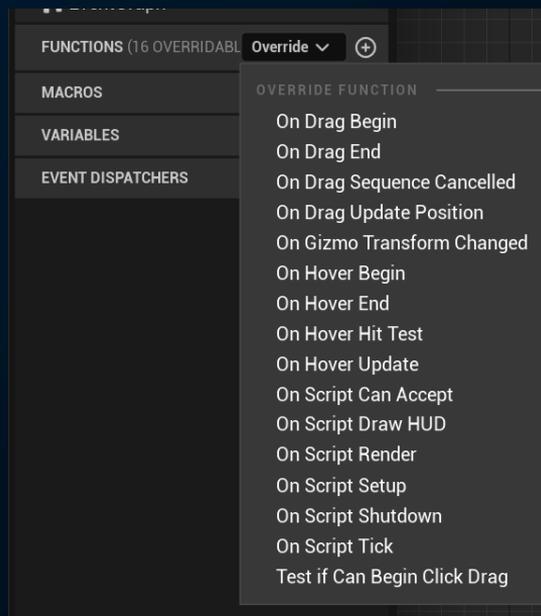
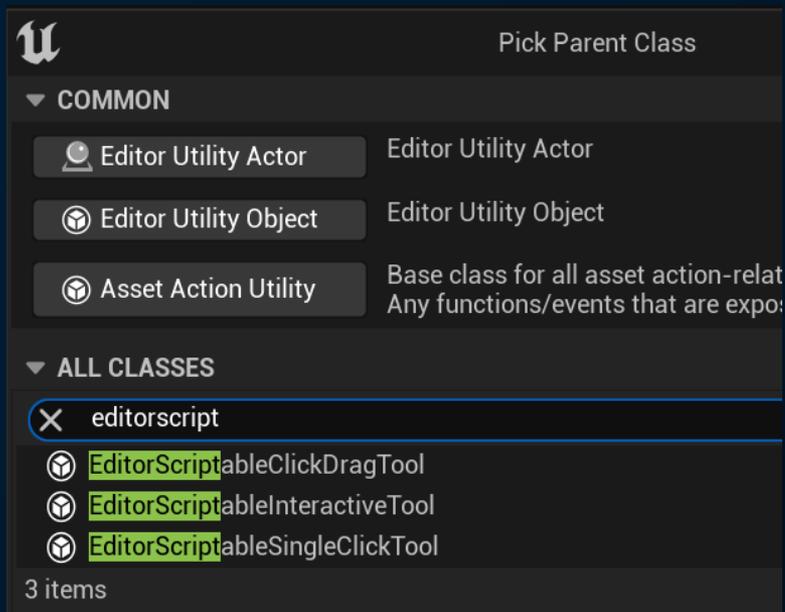
エディタ上で引いた線に沿って  
草・枝を配置 ([Twitter](#)より)



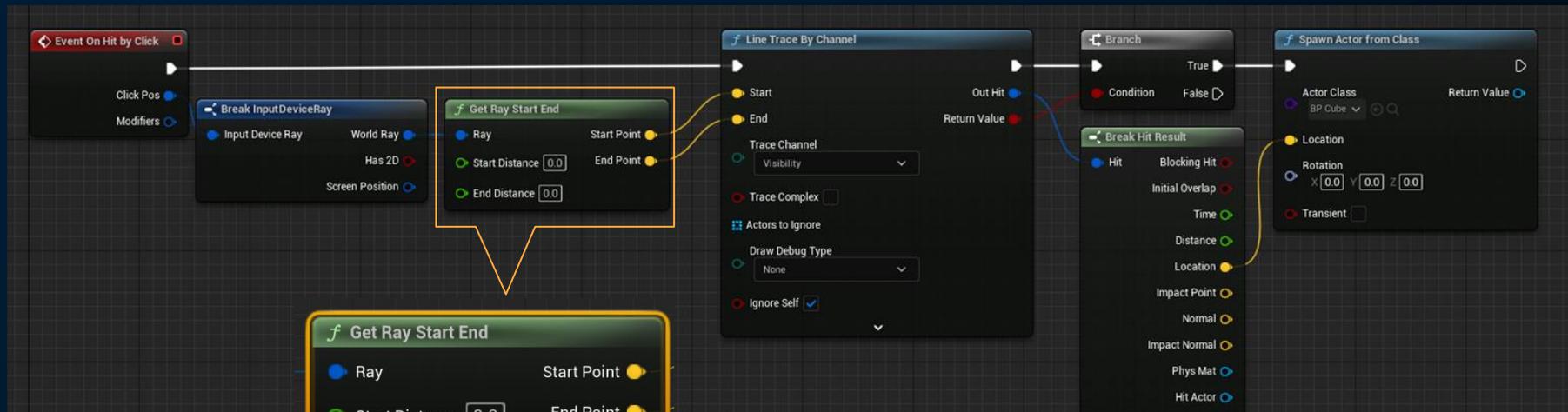
横スクロールゲーム用  
ステージエディタ ([Twitter](#)より)

# ツール作成用の EditorUtility Blueprint

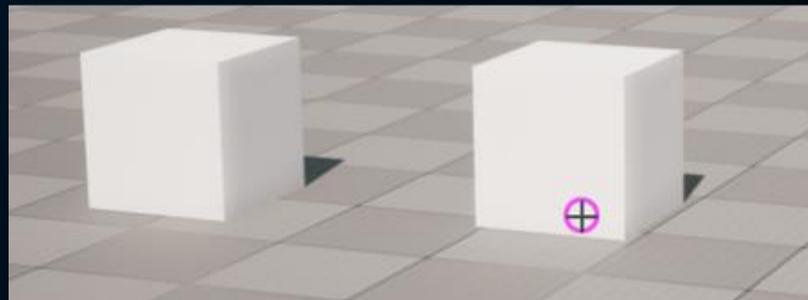
マウスの入力イベント、テキスト表示、3Dギズモの生成・制御など基本的な機能を標準で提供



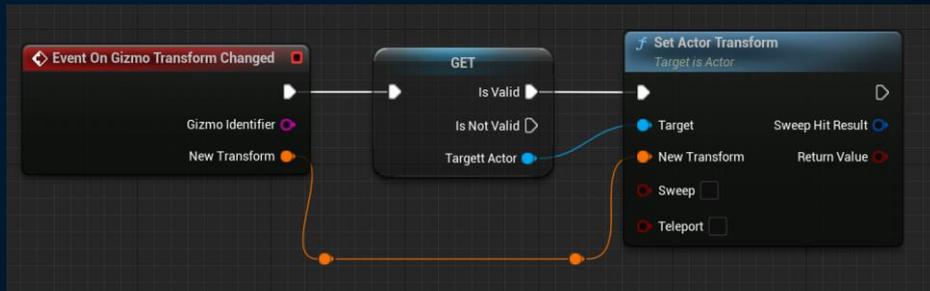
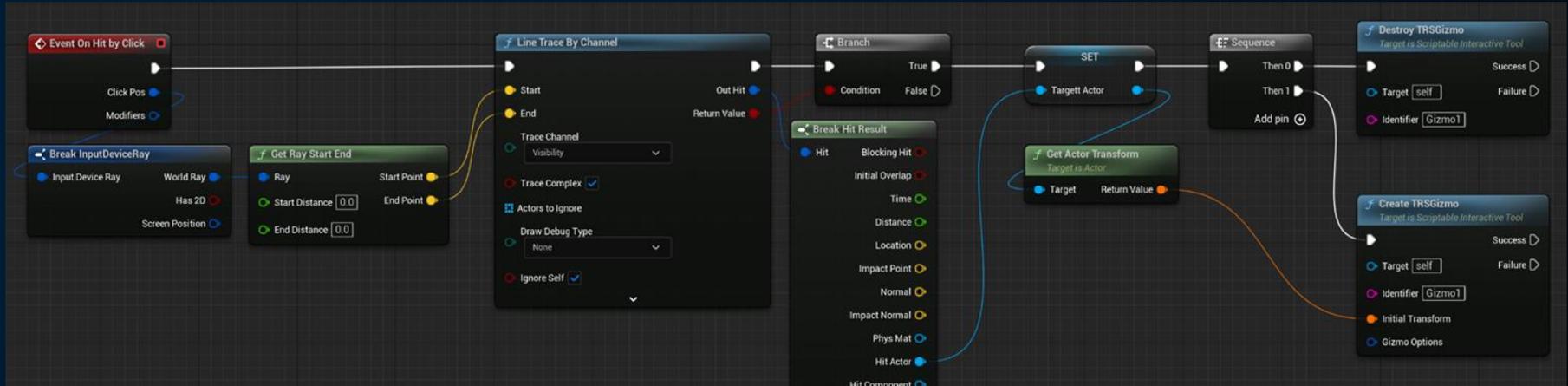
# マウスクリックした位置にActorを生成



GeometryScriptプラグインを有効にする必要あり

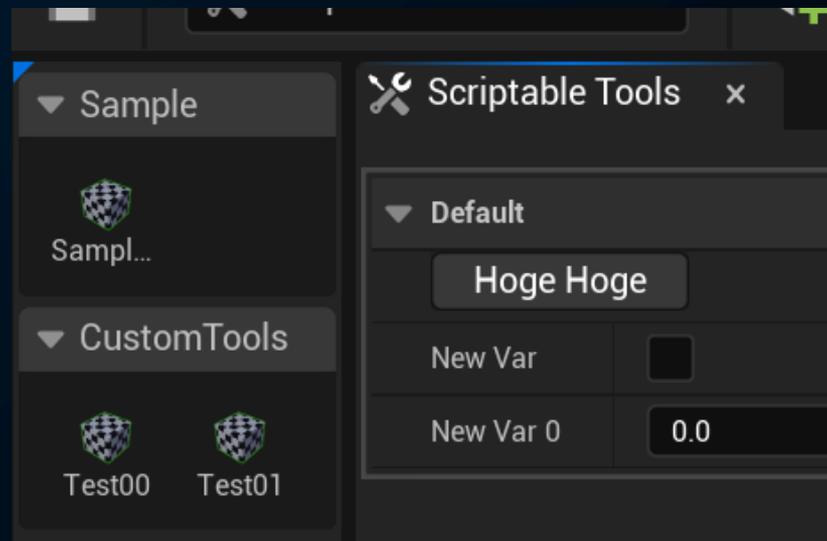


# 3DギズモによるActor制御

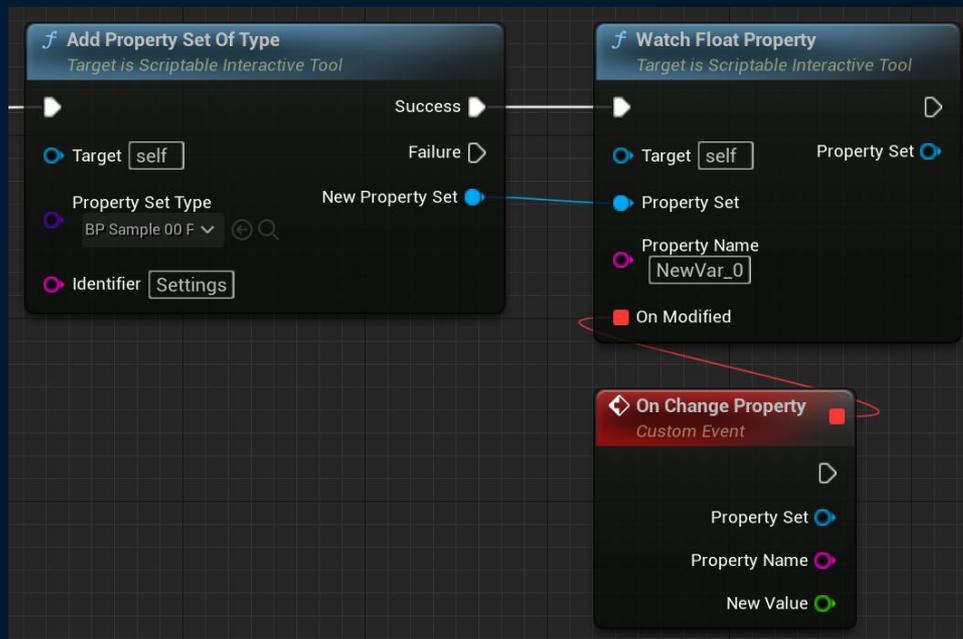


# ツールのプロパティ、イベントを公開

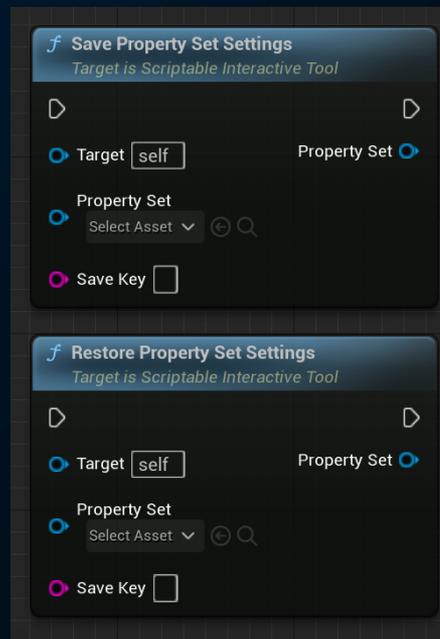
ScriptableInteractiveToolPropertySet 継承の  
BPを用意・登録すればOK！



# PropertySet を扱うためのヘルパー関数



プロパティの変更イベントを検出



プロパティの保存・復元

# ツールでの テキストメッセージ

**f Display User Warning Message**  
*Target is Scriptable Interactive Tool*

▶

Target

Message  📣

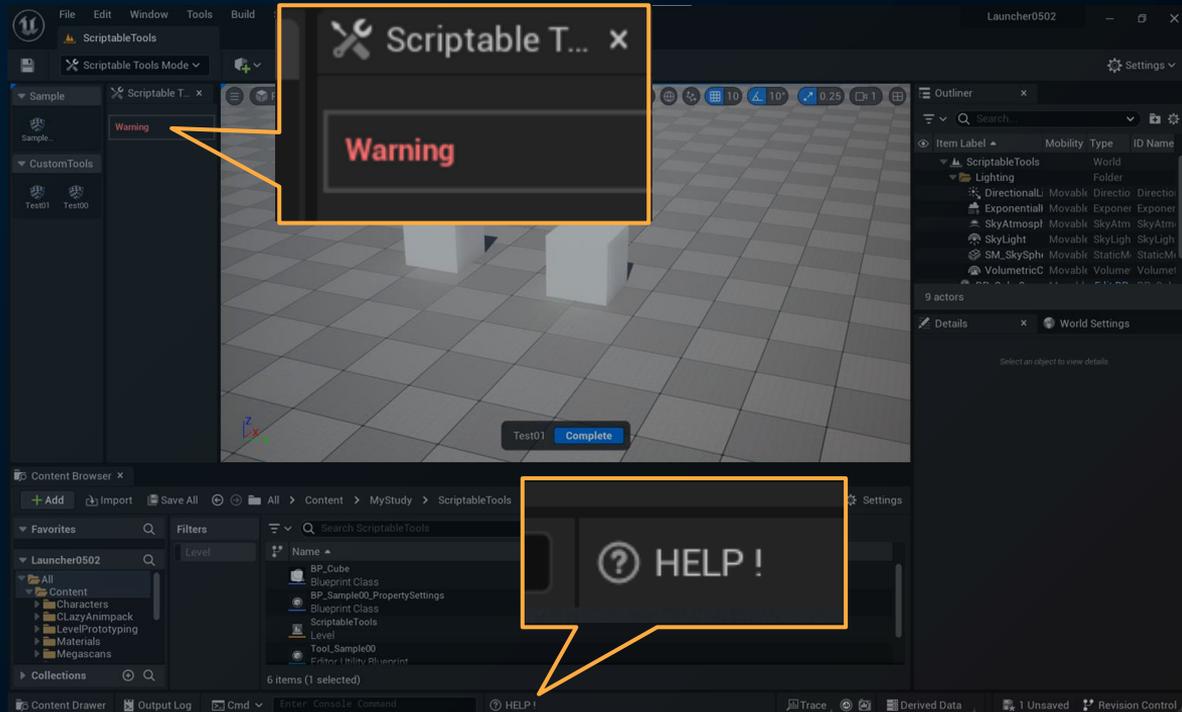
  

**f Display User Help Message**  
*Target is Scriptable Interactive Tool*

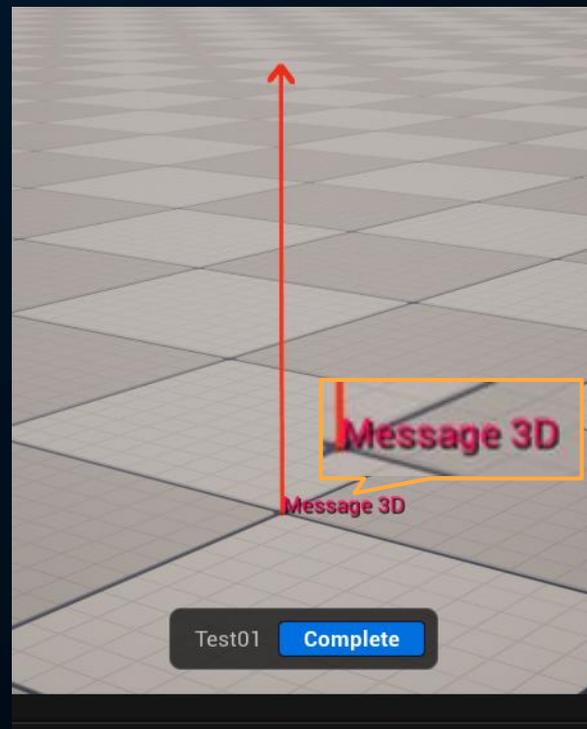
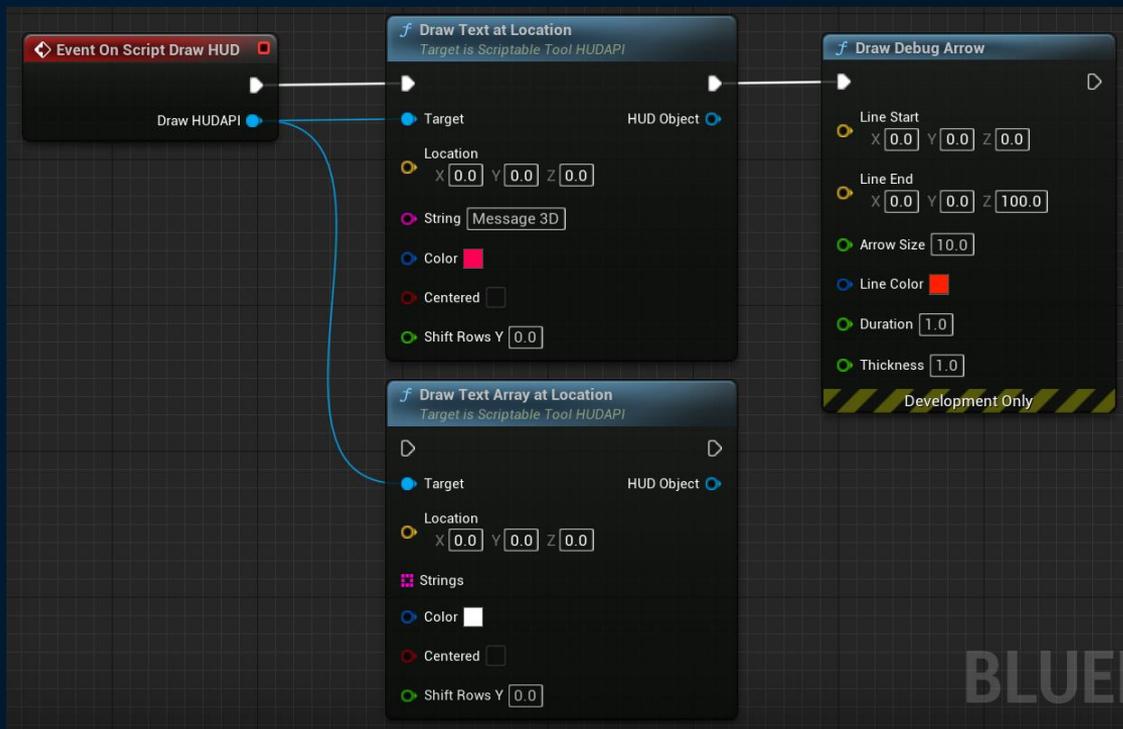
▶

Target

Message  📣



# ビューポートへのテキスト・デバッグ表示



# Scriptable Tools Framework はいいぞ！

## プロジェクトに特化したレベルデザインツールを作れる！

Geometry Script や

Procedural Content Generation Framework などと組み合わせたり

## ビューポートでの操作を限定できるという強みも！

他ツール（選択など）が無効化される

意図しない編集によるトラブル・ヒューマンエラーを回避できる

ご紹介した通り、**ツールで必要な機能をシンプルな形式で提供**

その反面、カスタムUIをつくるのが現時点でできない

**EditorUtilityWidget**とうまく使い分け・併用するなどの運用が必要

# 学習リソース

## Scriptable Tools & Editor Mode

<https://dev.epicgames.com/community/learning/tutorials/1loo/unreal-engine-scriptable-tools-editor-mode>

## Scriptable Tools & Editor Mode Reference

<https://dev.epicgames.com/community/learning/tutorials/7BKd/unreal-engine-scriptable-tools-system-editor-mode-reference>

## Scriptable Tools導入の流れ (Twitter)

<https://twitter.com/pafuhana1213/status/1654124259684519938>

## Editor Scriptable Tools: Basic Tools UE5.2

<https://www.youtube.com/watch?v=xWxtxS2jADo>

## UE5.2 Scriptable Tools : Node overview

<https://www.youtube.com/watch?v=w-PXmo8QKZs>

## Unreal Engine 5.2 - Scriptable Tools Introduction (Physics Scattering)

<https://www.youtube.com/watch?v=0-DcPwR4Cl0>



**UNREAL**  
ENGINE

**Mobile / Mac**

**UE5.2 アップデート**

# モバイル プレビューの改善点

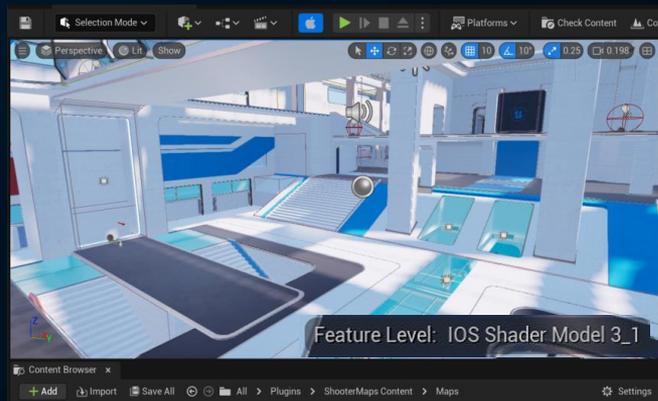
# モバイルプレビュー

異なるモバイルデバイス（レンダラー）における描画結果の違いをエディタ上で **エミュレート** する機能

<https://docs.unrealengine.com/5.2/ja/using-the-mobile-previewer-in-unreal-engine/>



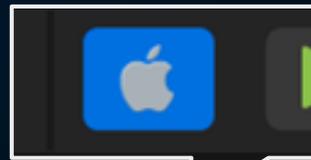
D3D Shader Model 6



iOS Shader Model 3.1

Settings

このボタンで  
ON/OFFを切り替え可能



GAME SPECIFIC SETTINGS

- World Settings
- Project Settings...
- Plugins

SELECTION

- Allow Translucent Selection
- Allow Group Selection
- Strict Box Selection
- Box Select Occluded Objects
- Show Transform Widget
- Show Subcomponents

SCALABILITY

- Engine Scalability Settings
- Material Quality Level
- Preview Rendering Level**

RENDERING

Volume

SNAPPING

Mobile preview using iOS material quality settings.

- Enable Socket Snapping
- Enable Vertex Snapping
- Enable Planar Snapping

VIEWPORT

- Hide Viewport UI
- Previewing

PREVIEW DEVICES

- Android ES 3\_1
- Android Vulkan
- IOS Shader Model 3\_1**
- Vulkan PC Shader Model 5
- D3D Shader Model 5
- D3D Shader Model 6 (Disable Preview)
- D3D Shader Model 3\_1



# UE5.2 リリースノートより

## モバイルプレビューの改善点

UE 5.2 では、Unreal Editor のモバイルプレビュー モードに安定性と精度の改善をいくつか行いました。これは、[Settings (設定)] メニューから [Preview Rendering Level (プレビュー レンダリング レベル)] のデバイスを選択することで使用可能です。

- プレビュー デバイスを変更する際の安定性を改善しました。
- エディタで [Engine Scalability Settings (エンジンの拡張機能設定)] ウィンドウを使用すると、プレビュー プラットフォームのスケラビリティ設定が適用され、プラットフォームでオーバーライドされたスケラビリティ変数がエディタ ビューポートに適用されるようになりました。たとえば、プラットフォームのスケラビリティ ini ファイルのオーバーライドに依存するメッシュ LOD トランジション距離のような設定が正確にプレビューされます。
- ターゲットのプレビュー プラットフォームのデバイス プロファイルによって設定されるコンソール変数がすべて適用されるようになるため、エディタ ビューポートでは、デバイス上で実行されると有効になる一連のレンダリング機能が正確にプレビューされます。

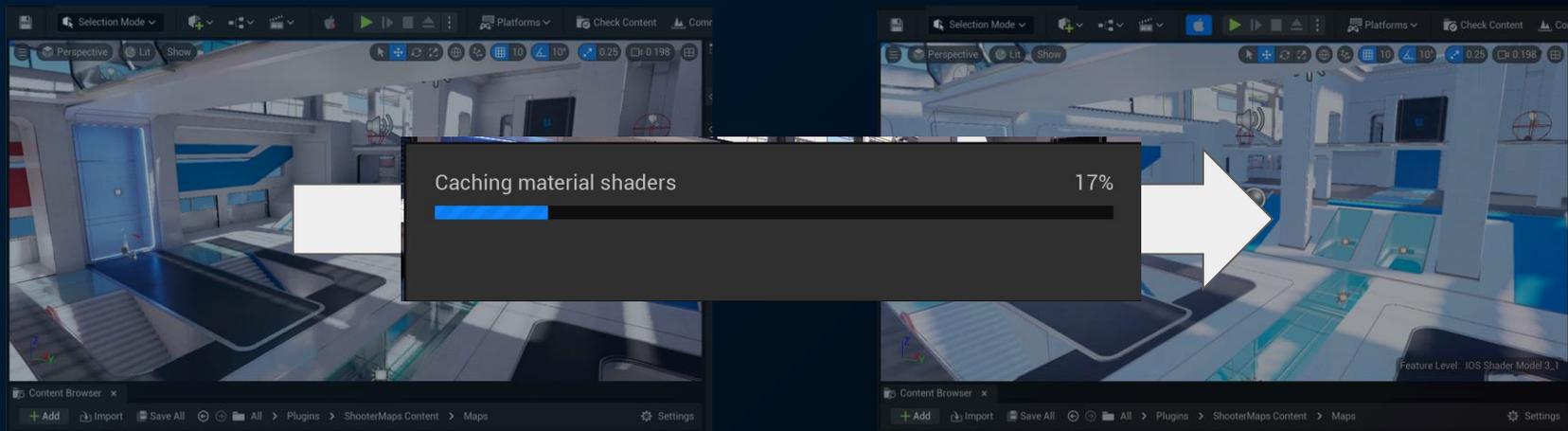
# モバイルプレビューへの切り替え時間が爆速に

ボタンを押してから  
モバイルプレビューに切り替わるまで

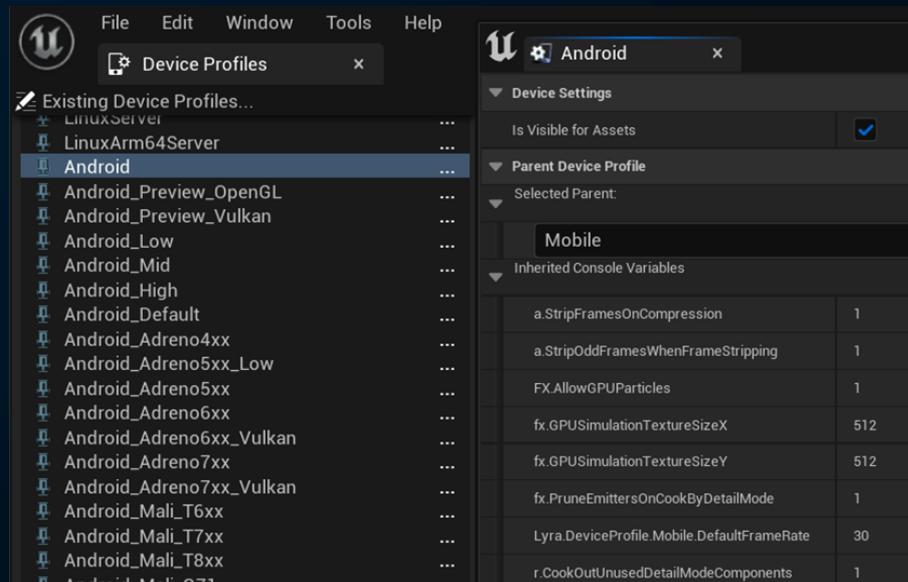
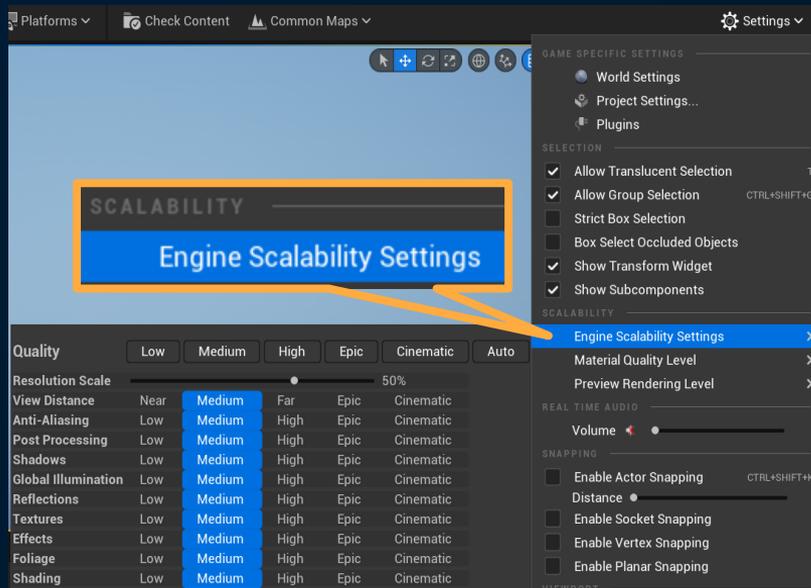
※ Lyra & 2回目以降での計測  
(初回はシェーダコンパイルがあるため)

UE5.1 : 17秒 **UE5.2 : 2秒**

大規模なプロジェクトの場合  
5~10分 から 数秒~1分に改善することも



# プラットフォーム毎の品質・機能ごとの設定を適用し より高精度なエミュレートを実現

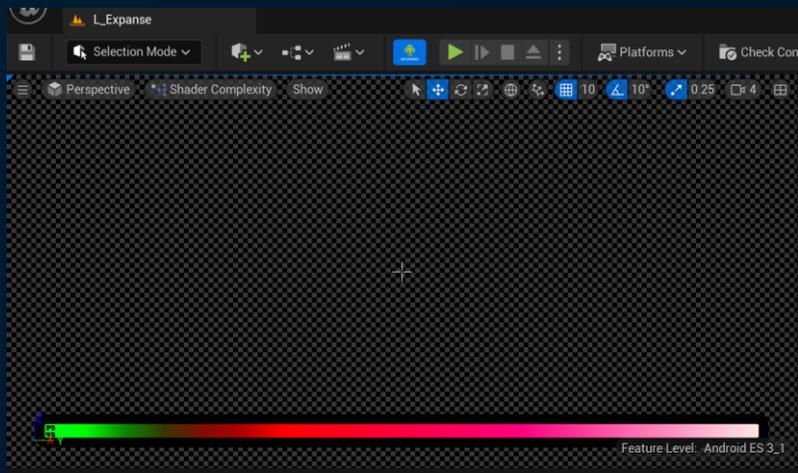


Scalability(機能ごとの品質設定)が  
プラットフォーム毎の管理に

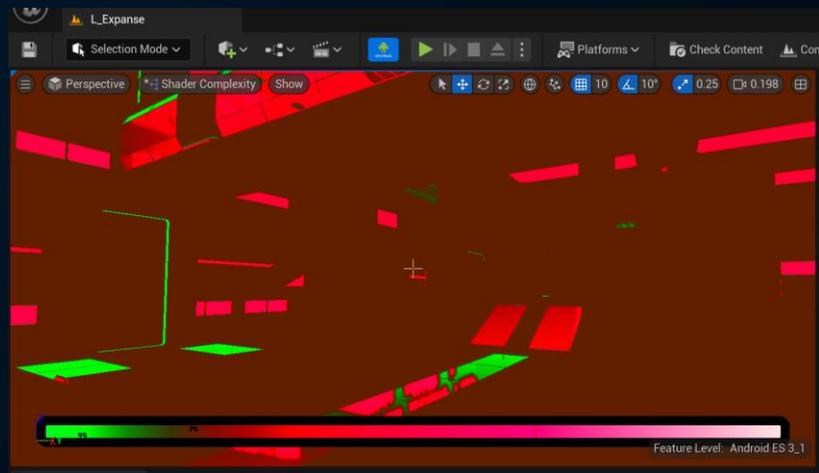
デバイスプロファイルで設定した  
CVar(コンソール変数)を適応するように

# 様々な不具合を修正し、安定性が向上

シェーダ複雑度（Shader Complexity）が  
モバイルプレビューでも正常に動作するように（5.3でさらに改善予定）



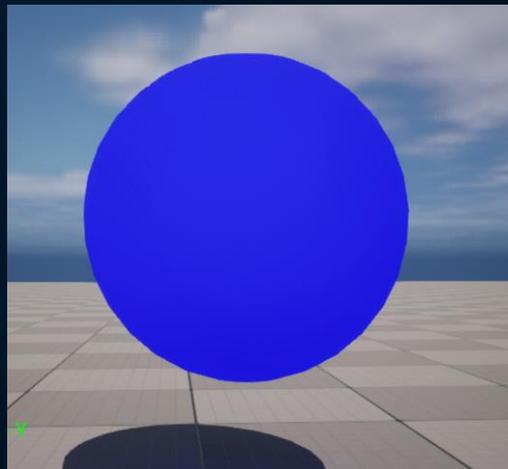
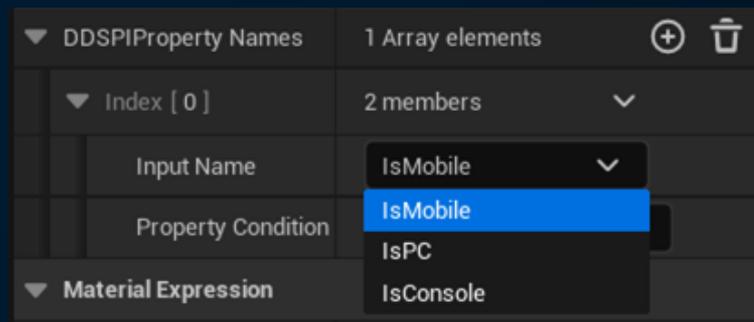
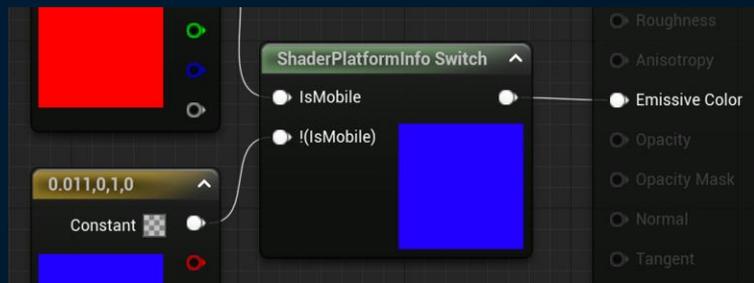
UE5.1



UE5.2

# ShaderPlatformInfoSwitch ノードを追加

プラットフォームの種類で Switch できるマテリアルノード



PC



Mobile

Beta

# iOS、tvOS、iPadOS の デバッグの準備 (Prepare For Debugging)

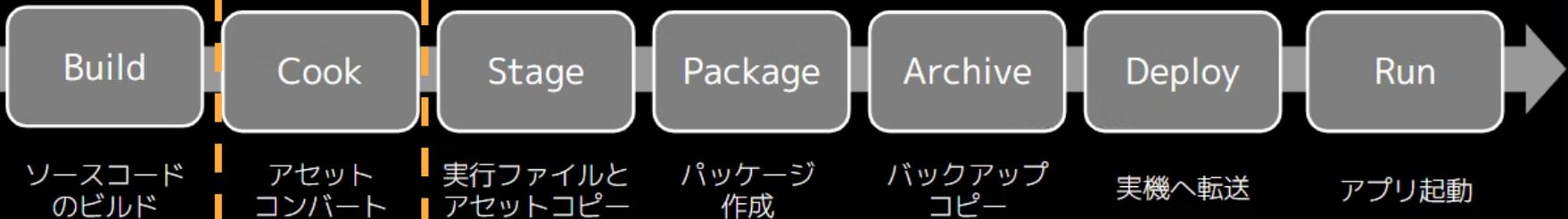
# Prepare For Debugging

過去のCook(アセットの変換)データを使い回すことで、  
パッケージの作成・デバッグを高速に行うための仕組み

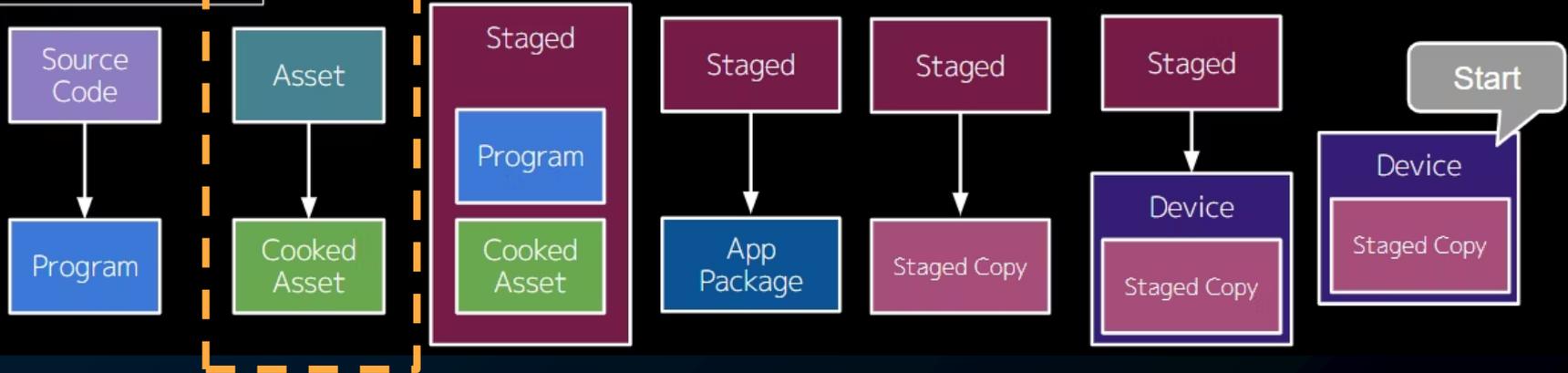


[\[UE4\] ビルドプロセスの流れ](#) より

# 処理フロー



# 処理イメージ



スマンがビルドを  
作ってくれんかニヤ  
マルチモードを  
試したいニヤ



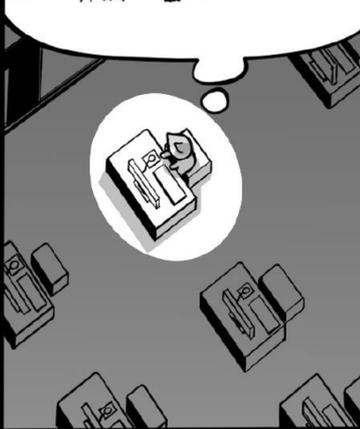
ニヤ!?  
今からですニヤ!?

急いで  
頼むニヤ



今日は野球を  
観に行く予定  
だったニヤ...

あ もう間に合わないニヤ...



## 最低限必要な作業

- Build (コードのビルド)
- Cook (アセットの変換)

## 最悪のケース

- エディタビルド  
(Cookするために)
- Build, Cook
- 失敗
- コード・アセットをDL
- エディタのビルド
- Build, Cook
- ...

「Press Button, Drink Coffee」

UE4における ビルドパイプラインとメンテナンスの全体像より

# Prepare For Debugging の効果

過去のCook結果を使い回すことで…

- Cook処理をスキップ
- エディタのビルドも不要に！



**ソースコードのビルドのみで  
パッケージ作成完了！**

# Prepare For Debugging の効果

過去のCook結果を使い回すことで…

- Cook処理をスキップ
- エディタのビルドも不要に！



ソースコードのビルドのみで  
パッケージ作成完了！



パッケージの実行・デバッグまで  
最長1日以上かかっていたのが…

**最短で数秒、長くても数分に！**



※ 2019年の画像

※ 横浜ファンの人が  
作りました

### 「Press Button, Drink Coffee」

UE4におけるビルドパイプラインとメンテナンスの全体像より

# Prepare For Debuggingの使い方・今後の予定

UATの

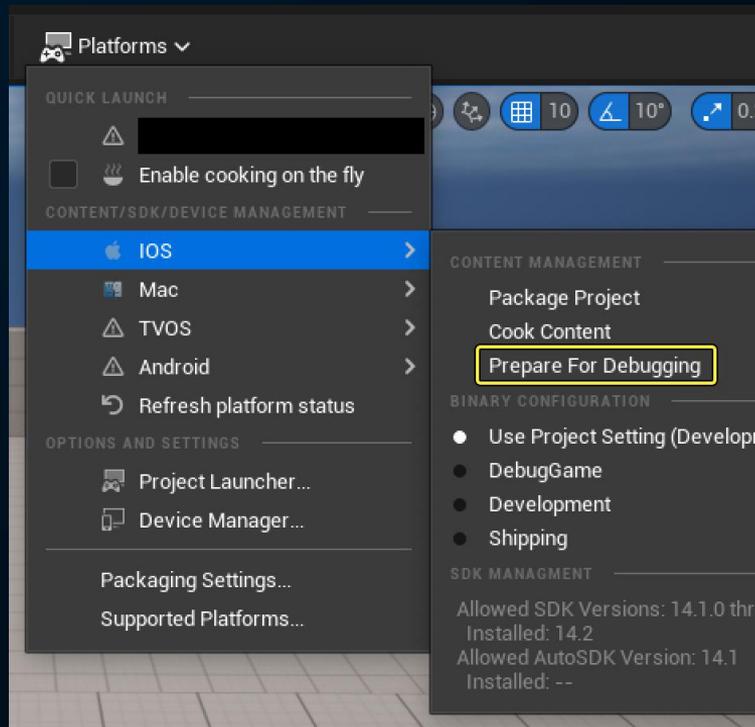
**WrangleContentForDebugging** コマンド  
または、エディタから実行可能

RunUAT.command

```
WrangleContentForDebugging -platform=ios -  
ProjectFilePath=XXX.uproject  
-nocompile -nocompileuat
```

UE5.3 で

Windowsからのデバッグにも対応予定！

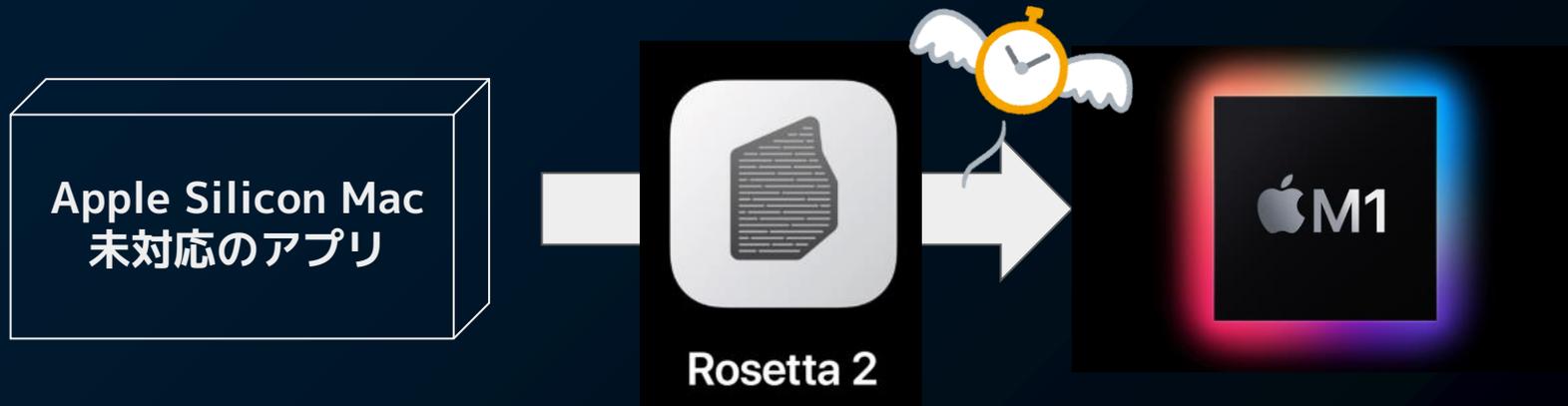


# Unreal Editor での Apple Silicon のネイティブ サポート

# Intel Mac と Apple Silicon Mac

2020年以降よりIntel製のSoCではなく、**Apple独自のSoC (Apple Silicon : M1, M2)** を積んだMacが販売されるように

Intel Mac時代に開発されたアプリを  
Apple Silicon Mac で動かすためには、**Rosetta 2 によるエミュレート**が必要



# UE5 の Apple Silicon対応状況

## UE4.27, UE5.0 :

Apple Silicon対応のパッケージを作成可能に！  
Unreal Editorは未対応

## UE5.1 :

Unreal Editor（ビルド版）が対応！  
Unreal Editor（ランチャー版）は未対応

## UE5.2 :

**Unreal Editor（ランチャー版）も対応！**  
マーケットプレイスも対応



いまここ！

# ネイティブサポートによるCPU負荷の削減

**CPU負荷が 3~5 ms 削減!**

プロジェクト : Lyra 環境 : MacBookPro 2021 ( M1 Max )



UE5.1 TAA Game : 9 ~ 10 ms  
SkyLightの不具合あり (5.2で修正済み)



UE5.2 TAA Game : 5 ~ 8 ms

# UE 5.2 における macOS の制限事項

- **Nanite は M2 Mac のみ実験的にサポート**  
デフォルトは無効。有効にするためには [コード変更](#) が必要
- **ヘア (毛髪)/毛皮/グルームのストランド は未サポート**
- ハードウェア レイトレーシングは非対応のため  
**Lumen は ソフトウェアレイトレーシングのみ使用可能**
- TSR の最適化が完了していないため、  
**アンチエイリアシング設定を FXAAまたはTAAに変更することを強く推奨**
  - TSR -> TAA でGPU負荷が 10ms 改善

# M2 Nanite

## M2 デバイスで実験的な Nanite サポートを使用する方法

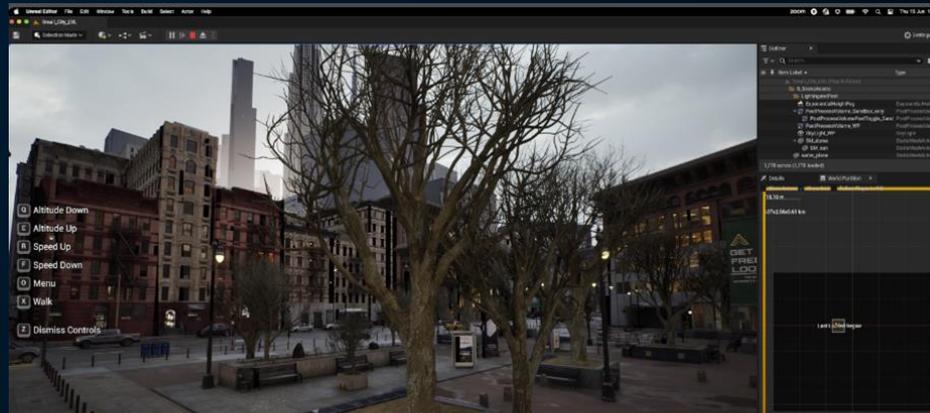
待望の macOS での Nanite の完全サポートに向けた最初のステップとして、ue5-main および GitHub 上の 5.2 ブランチで実験的サポートが利用できるようになりました。ただし、これはデフォルトでは無効になっています。これを有効にするには、Unreal Engine のソースコードで次の変更を行う必要があります。

1. 「UEBuildMac.cs  
[Engine/Source/Programs/UnrealBuildTool/Platform/Mac/UEBuildMac.cs]」で PLATFORM\_MAC\_ENABLE\_EXPERIMENTAL\_NANITE\_SUPPORT=1 を設定します。
2. 「spirv\_msl.hpp」で UE\_EXPERIMENTAL\_MAC\_NANITE\_SUPPORT を有効にします。
3. 「UEBuildMac.cs」の指示に従って ShaderConductor をリビルドします。
4. 「DataDrivenPlatformInfo.ini [Engine/Config/Mac/DataDrivenPlatformInfo.ini]」で bSupportsNanite=true と bSupportsUInt64ImageAtomics=true を設定します。

<https://www.unrealengine.com/ja/tech-blog/unreal-engine-5-2-brings-native-support-for-apple-silicon-and-other-developments-for-macos>



# UE5.3 から M2 Macにおける Naniteサポートが Production-Readyに



## Support for Nanite on Apple M2 Devices



THOMAS CONVARD | Posted on June 8

Unreal Engine 5.3 brings Production-Ready support for Nanite rendering technology on Apple Silicon M2 devices. This support is enabled by default in the UE macOS binaries downloadable from the Epic Games launcher, whereas the Experimental 5.2 support was only available when building from github sources.

<https://portal.productboard.com/epicgames/1-unreal-engine-public-roadmap/c/1151-support-for-nanite-on-apple-m2-devices>

# Appleプラットフォーム向けのUE開発環境 (2023/6)

UE5.1 ~ 5.2 で多くの改善が入り  
1~2年前と比べると、Macを使った開発が凄く快適に！

今から開発環境を整えるなら

- M1 ではなく、M2 系のチップ搭載で
- メモリは 16 GB 以上に
- 負荷が低いシーンなら、MacBook Air でもOK  
高負荷なら、MacBook Pro 以上で



Thanks Axel !

モバイル開発向けの機能追加・改良は  
UE5.3以降もガンガン入る予定です

ご期待下さい！

鈴木パートへ



**UNREAL**  
ENGINE

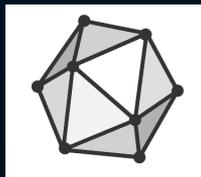
# **NNE plugin**

Neural Network Engine

# NNE プラグイン

## 目的

- ✓ 学習済みの**ONNXファイル**をハイパフォーマンスで実行
- ✓ クロスプラットフォームで動作 (デスクトップ + 現世代コンソール機)
- UE5.0で導入されたNNIプラグインが改名されたもの
- CPUまたはGPUで動作するが一部プラットフォームは現時点ではCPUのみ



<https://onnx.ai/>

# NNE プラグイン 学習用コンテンツ

Home / Unreal Engine / Learning / Epic Games / NNE - Quick Start Guide

## NNE - Quick Start Guide

Learn how to program a C++ class that is able to run a neural network and how to use it from a blueprint.

by Nice  Feb 25, 2023 • Last Updated: May 15, 2023 • Applications:  •  For Beginners

BEI TUTORIAL

5    840 Views  Add to Favorites

### NNE - Quick Start Guide

#### Introduction

In this tutorial you will program a small C++ class that can be exposed to actors to run an arbitrary neural network on CPU from within a blueprint.

#### Goals

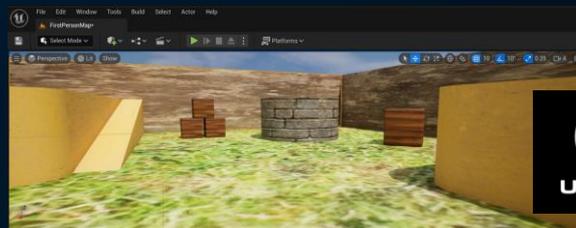
In this tutorial you will...

- ... setup a simple C++ project
- ... program a function that loads a neural network from an asset
- ... feed the neural network with blueprint typed in- and output tensors
- ... add a function to run the inference
- ... setup a blueprint actor to use the class

By the end of this tutorial you will be familiar with the key concepts of NNE and able to transfer the knowledge to your own game.

**ON THIS PAGE**

- NNE - Quick Start Guide
- Introduction
- Goals
- Prerequisites
- Project Setup**
- C++
- Setup
- Implementation
- Runtime Enumeration
- Model Creation
- Tensor Creation
- Model Queries
- Input Configuration
- Model Execution
- Blueprint
- Setup
- Implementation
- Model Loading



<https://dev.epicgames.com/community/learning/tutorials/yXJ3/unreal-engine-nne-quick-start-guide>

<https://github.com/microsoft/OnnxRuntime-UnrealEngine>

# NNE プラグイン



NNE **実験段階**

Cross-platform framework for running deep learning and neural network inference in Unreal Engine, supporting both CPU and GPU acceleration.



編集



パッケージ

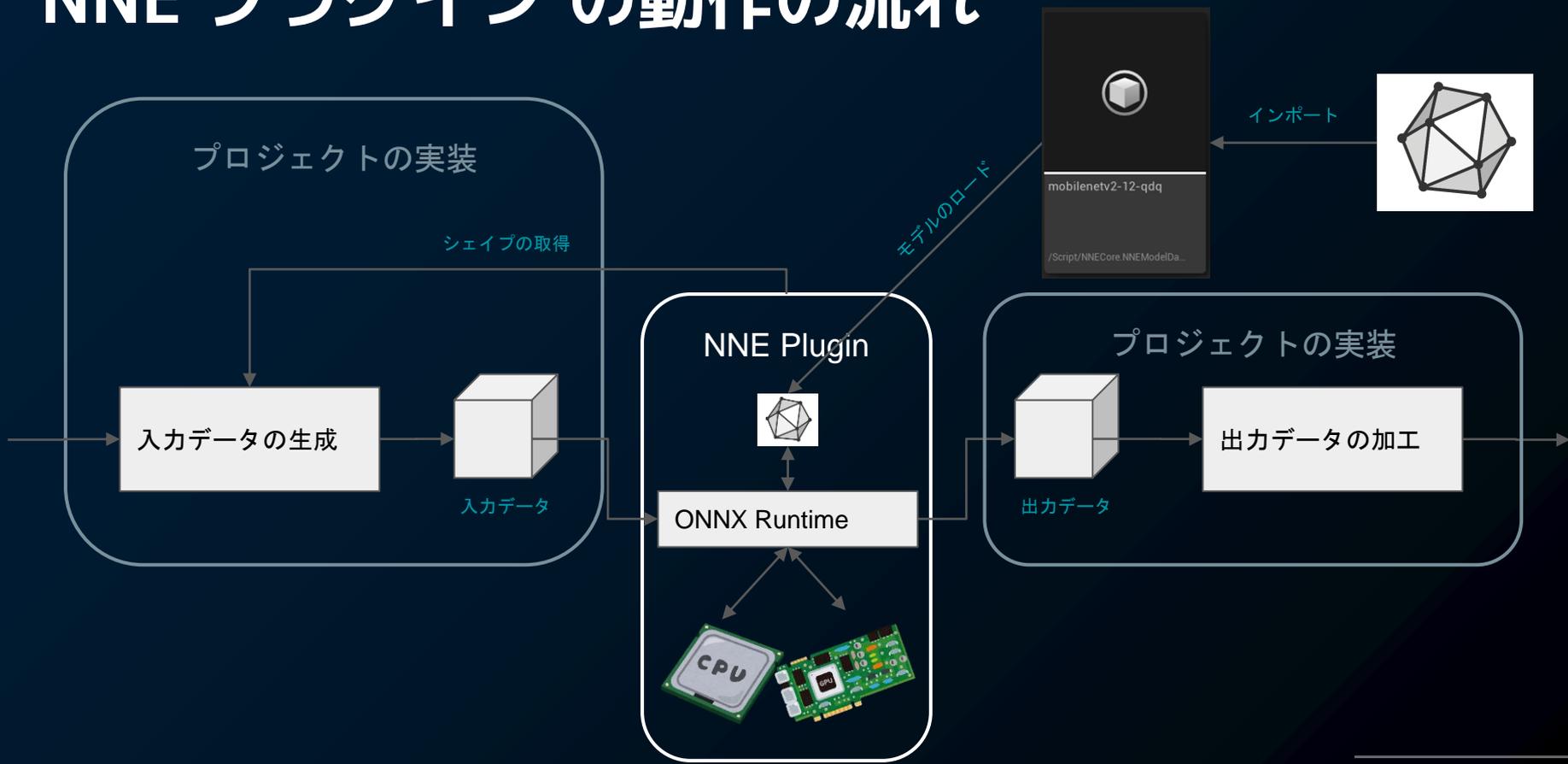
バージョン 0.1

Epic Games, Inc.

## Experimental



# UNE プラグイン の動作の流れ



# 実コードサンプル

```
bool UNeuralNetworkModel::SetInputs(const TArray<FNeuralNetworkTensor>& Inputs)
{
    check(Model.IsValid())

    using namespace UE::NNECore;

    InputBindings.Reset();
    InputShapes.Reset();

    TConstArrayView<FTensorDesc> InputDescs = Model->GetInputTensorDescs();
    if (InputDescs.Num() != Inputs.Num())
    {
        UE_LOG(LogTemp, Error, TEXT("Invalid number of input tensors provided"));
        return false;
    }

    InputBindings.SetNum(Inputs.Num());
    InputShapes.SetNum(Inputs.Num());
    for (int32 i = 0; i < Inputs.Num(); i++)
    {
        InputBindings[i].Data = (void*)Inputs[i].Data.GetData();
        InputBindings[i].SizeInBytes = Inputs[i].Data.Num() * sizeof(float);
        InputShapes[i] = FTensorShape::MakeFromSymbolic(
            FSymbolicTensorShape::Make(Inputs[i].Shape)
        );
    }

    if (Model->SetInputTensorShapes(InputShapes) != 0)
    {
        UE_LOG(LogTemp, Error, TEXT("Failed to set the input shapes"));
        return false;
    }

    return true;
}
```

```
bool UNeuralNetworkModel::RunSync(UPARAM(ref) TArray<FNeuralNetworkTensor>& Outputs)
{
    check(Model.IsValid());

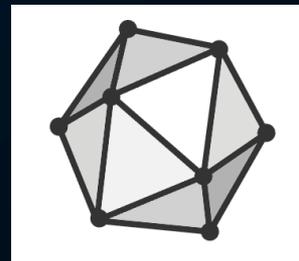
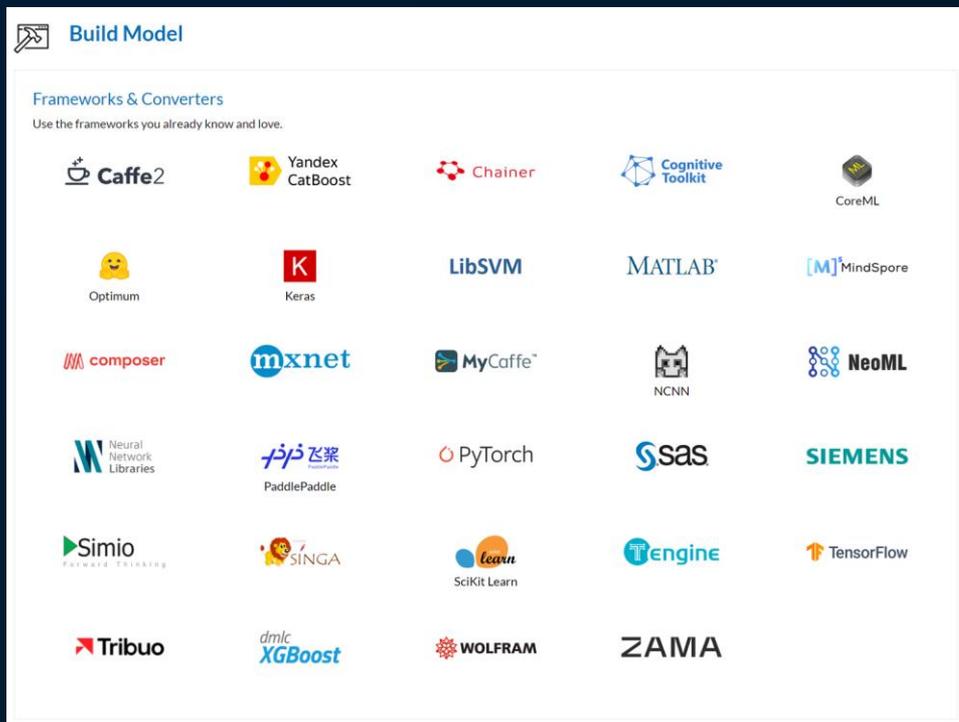
    using namespace UE::NNECore;

    TConstArrayView<FTensorDesc> OutputDescs = Model->GetOutputTensorDescs();
    if (OutputDescs.Num() != Outputs.Num())
    {
        UE_LOG(LogTemp, Error, TEXT("Invalid number of output tensors provided"));
        return false;
    }

    TArray<FTensorBindingCPU> OutputBindings;
    OutputBindings.SetNum(Outputs.Num());
    for (int32 i = 0; i < Outputs.Num(); i++)
    {
        OutputBindings[i].Data = (void*)Outputs[i].Data.GetData();
        OutputBindings[i].SizeInBytes = Outputs[i].Data.Num() * sizeof(float);
    }

    return Model->RunSync(InputBindings, OutputBindings) == 0;
}
```

# ONNXファイルは各種学習フレームワークから



<https://onnx.ai/supported-tools.html#buildModel>

# ゲーム内容を学習させたい！



**ML Adapter** Beta

[EXPERIMENTAL] A framework for training and utilizing machine learning agents in games. Creates an RPC interface through which an external process can query game state and control in-game actors. Once trained, agents can be run in-engine via neural networks loaded from ONNX models.

Edit Package

Version 0.0.1  
Epic Games, Inc.

**Very Experimental**

続報をお待ちください。

# 参考リンク

- [UE5] Neural Network Inferenceプラグインについて  
<https://historia.co.jp/archives/26238/>



**UNREAL**  
**ENGINE**

# Chaos Flesh

# ChaosFlesh

## 目的

- ✔ 組み込みの**軟体シミュレーション**
- ✔ エンジン内で編集
- 四面体メッシュによるシミュレーション
- シミュレーション結果を表示用メッシュに転送
- デバッグ用に四面体ボリュームを表示可能



# Chaos Flesh プラグイン



Chaos **Flesh** Beta

Chaos **Flesh** Simulation



Edit



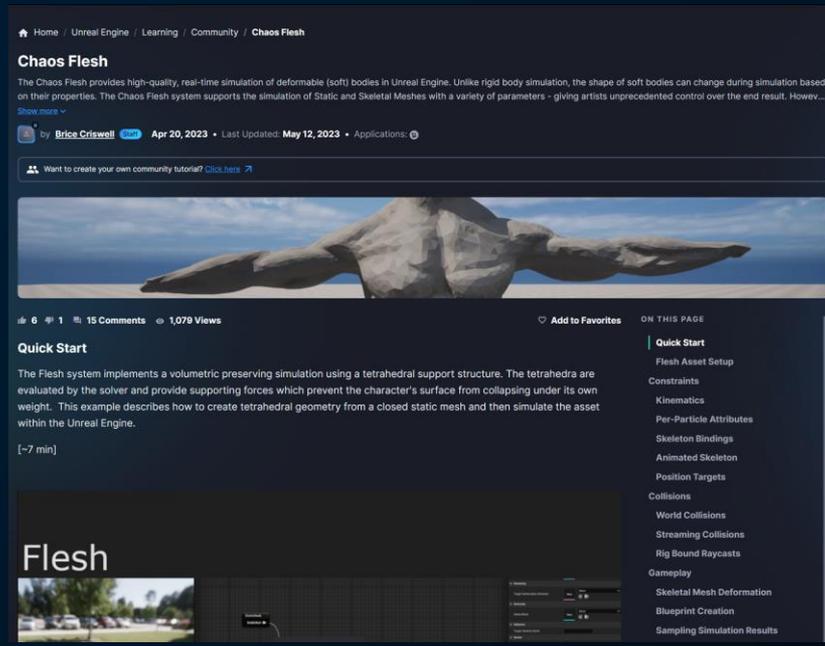
Package

Version 0.1

Epic Games, Inc.

**Experimental**

# Chaos Flesh チュートリアル



The screenshot shows a community tutorial page for "Chaos Flesh" on the Unreal Engine website. The page title is "Chaos Flesh" and it is authored by Brice Criswell. The tutorial description states: "The Chaos Flesh provides high-quality, real-time simulation of deformable (soft) bodies in Unreal Engine. Unlike rigid body simulation, the shape of soft bodies can change during simulation based on their properties. The Chaos Flesh system supports the simulation of Static and Skeletal Meshes with a variety of parameters - giving artists unprecedented control over the end result. However..."

Metadata: by Brice Criswell, Apr 20, 2023, Last Updated: May 12, 2023, Applications: 0.

Engagement: 6 likes, 1 share, 15 Comments, 1,079 Views.

Quick Start section: "The Flesh system implements a volumetric preserving simulation using a tetrahedral support structure. The tetrahedra are evaluated by the solver and provide supporting forces which prevent the character's surface from collapsing under its own weight. This example describes how to create tetrahedral geometry from a closed static mesh and then simulate the asset within the Unreal Engine." (~7 min)

Table of Contents (ON THIS PAGE):

- Quick Start
- Flesh Asset Setup
- Constraints
- Kinematics
- Per-Particle Attributes
- Skeleton Bindings
- Animated Skeleton
- Position Targets
- Collisions
- World Collisions
- Streaming Collisions
- Rig Bound Raycasts
- Gameplay
- Skeletal Mesh Deformation
- Blueprint Creation
- Sampling Simulation Results

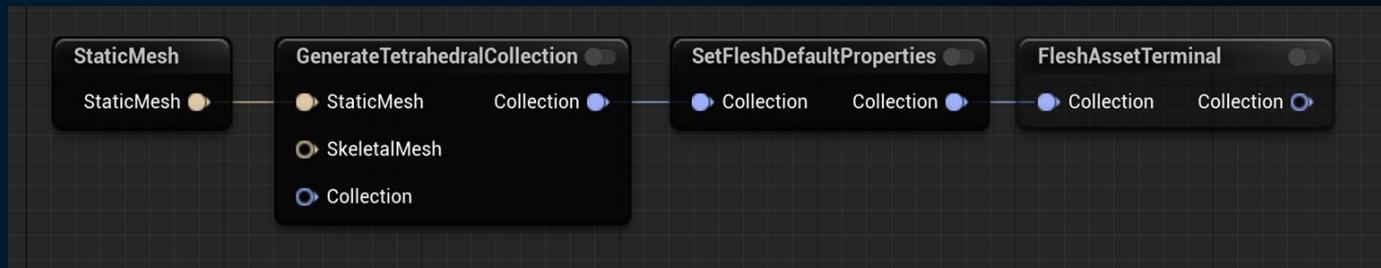
<https://dev.epicgames.com/community/learning/tutorials/BEby/unreal-engine-chaos-flesh>

# Chaos Flesh 構成

- DataGraph
  - 四面体メッシュ作成処理を記述したノードグラフ
- FleshAsset
  - DataGraphと使用するメッシュなどのパラメータ
  - 生成された四面体メッシュを保持
- DeformableSolverActor/DeformableSolverActor
  - シミュレーションソルバー
- FleshComponent
  - FleshAssetとDeformableSolverを指定

# DataGraph 概要

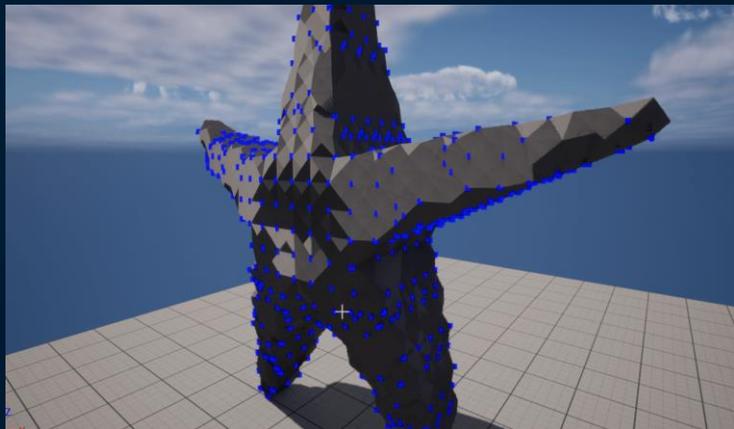
- シミュレーションメッシュ(四面体メッシュ)の作成
- シミュレーション頂点に値を設定
- コレクションをFleshAssetTerminal入力



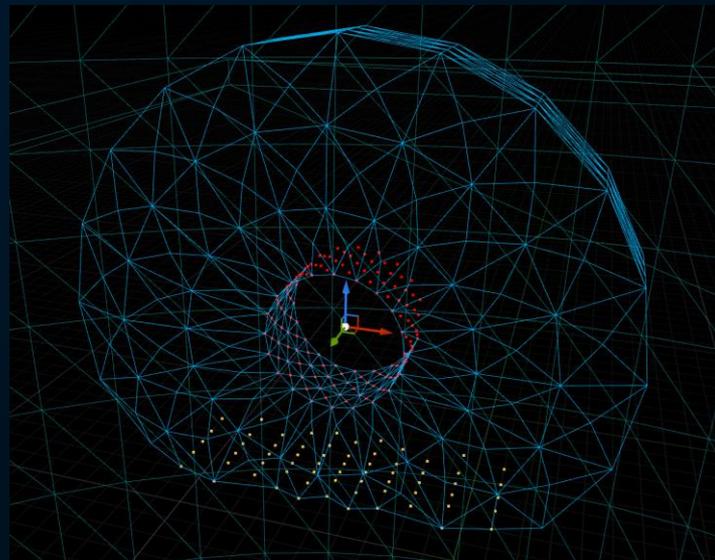
最小データグラフ



# DataGraph デバッグ用コンソールコマンド

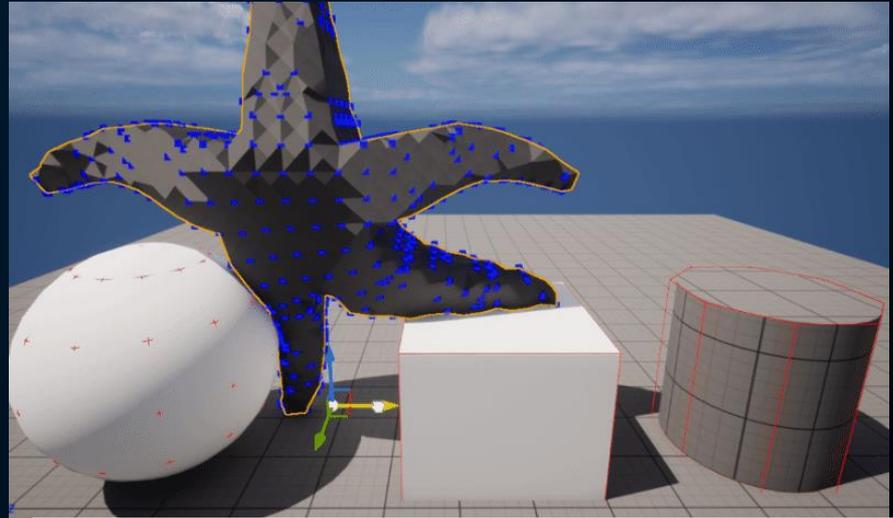
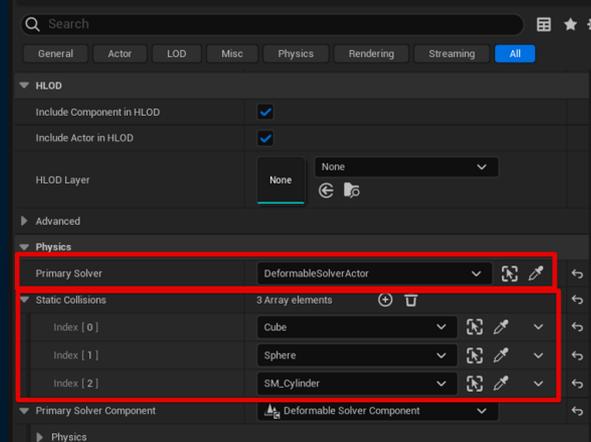
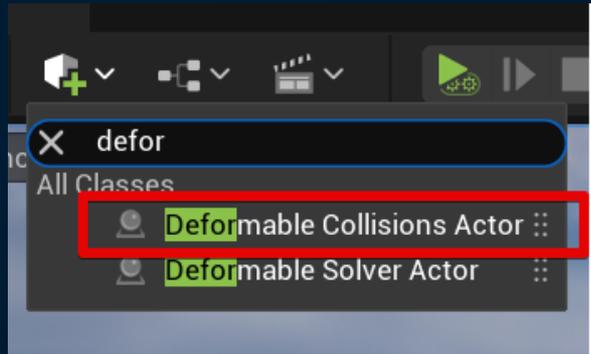


```
p.Chaos.DebugDraw.Enabled 1  
p.Chaos.DebugDraw.Deformable.TetrahedralParticle 1
```



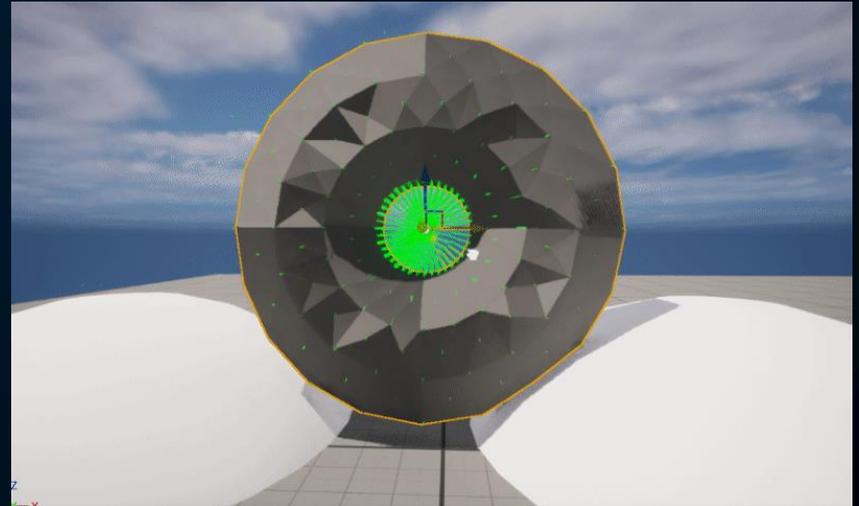
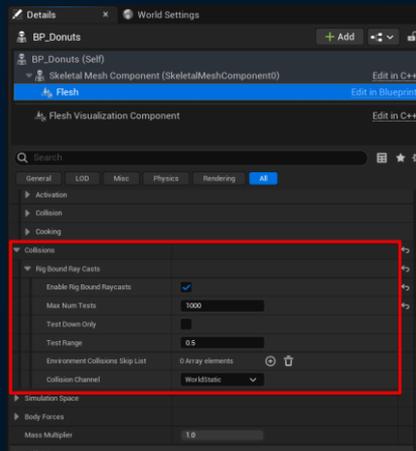
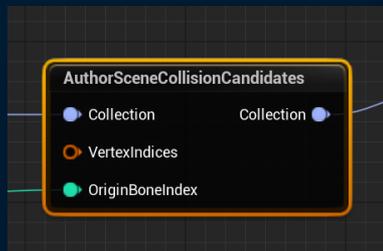
```
p.Chaos.DebugDraw.Deformable.KinematicParticle 1  
p.Chaos.DebugDraw.Deformable.TransientKinematicParticle 1
```

# DeformableCollision



p.Chaos.DebugDraw.Deformable.KinematicParticle 1  
p.Chaos.DebugDraw.Deformable.RigidCollisionGeometry 1

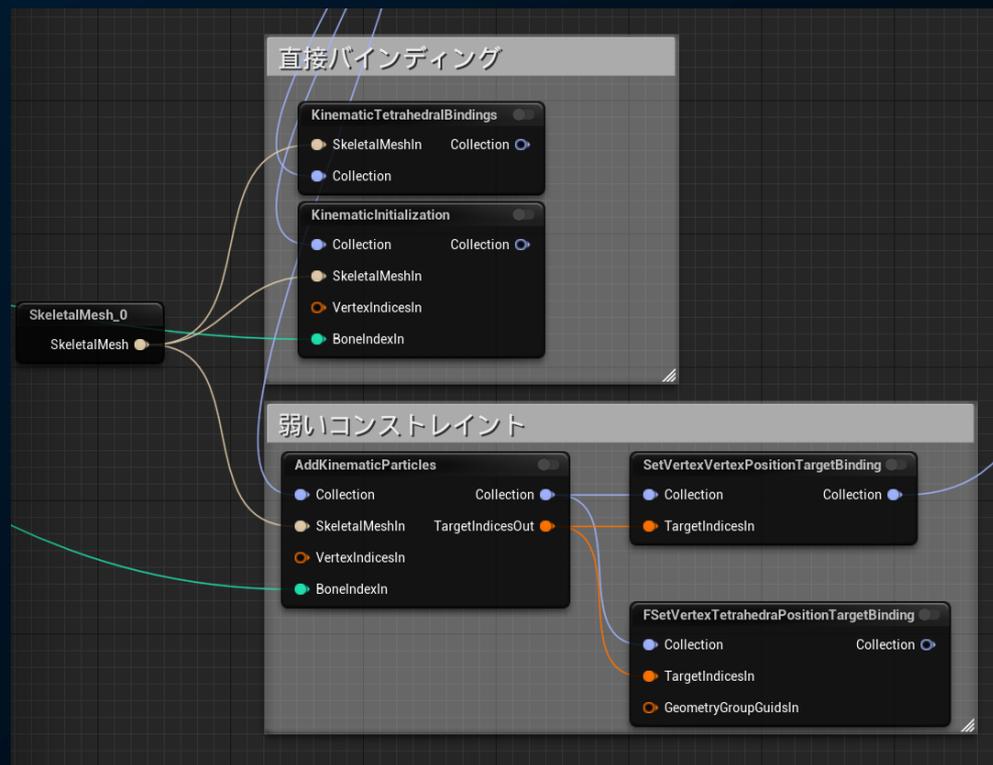
# RaycastCollision



p.Chaos.DebugDraw.Enabled 1  
p.Chaos.DebugDraw.Deformable.SceneRaycasts 1

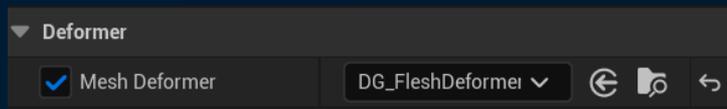
# アニメーションバインディング

- 親のSkeletalMeshComponentのアニメーションを取り込む
- 参照方法
  - 直接バインディング
  - ポジションターゲット

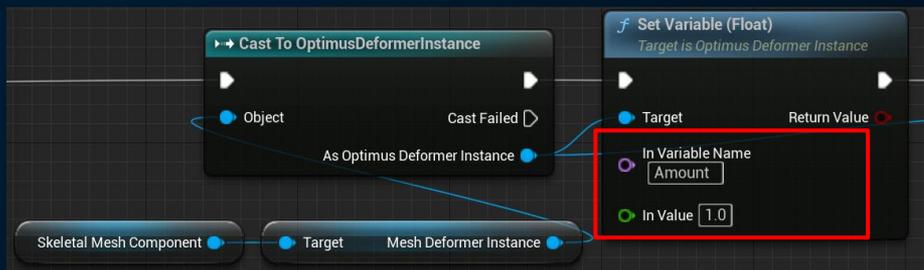


# SkeletalMeshで描画

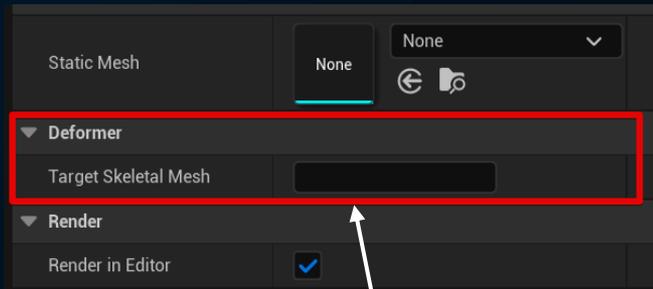
SkeletalMeshComponentのMeshDeformer設定



DeformerInstanceにAmountを設定



FleshAssetのTargetSkeletalMesh設定



Component名を指定  
SkeletalMeshが一つしかないなら空でもOK



**UNREAL**  
**ENGINE**

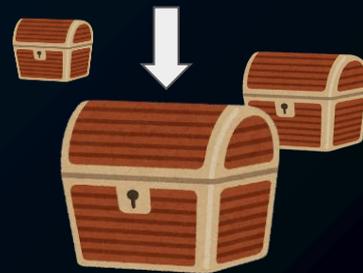
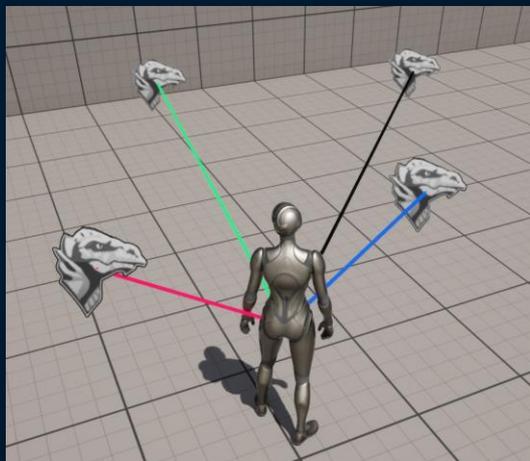
# Targeting System プラグイン

# Targeting System プラグイン

Beta

## 目的

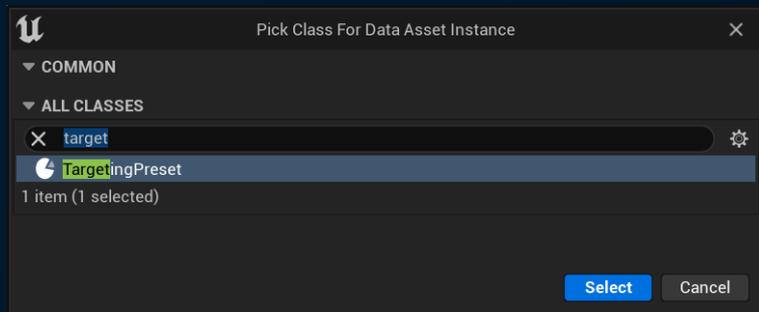
- ✔ 同期または非同期でアクターを検索する仕組み
- ゲームプレイアビリティまたはBP/C++から呼び出して利用



# TargetingPreset

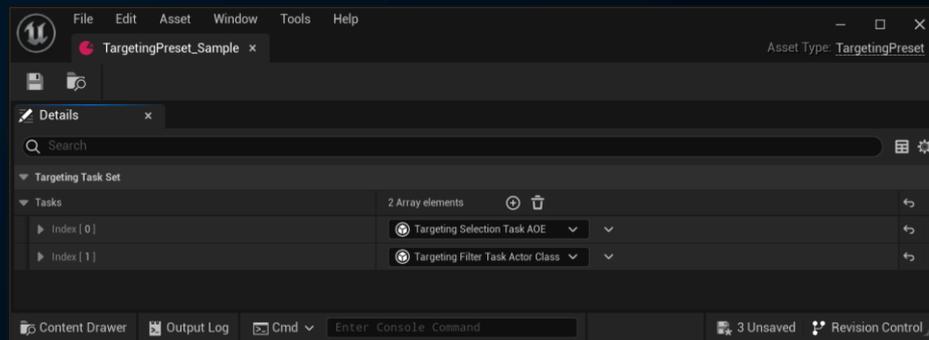
## ● TargetingPresetの作成

- **DataAssetから作成**する



## ● TargetingPresetの設定

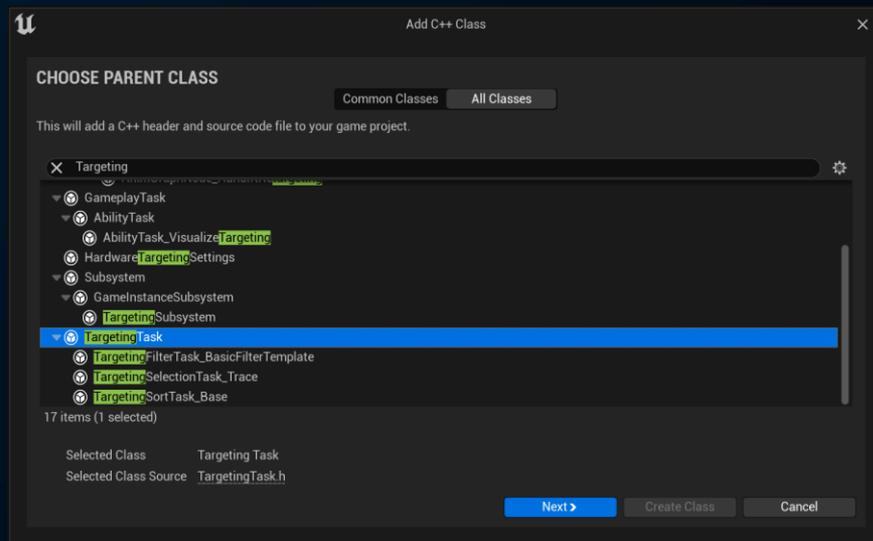
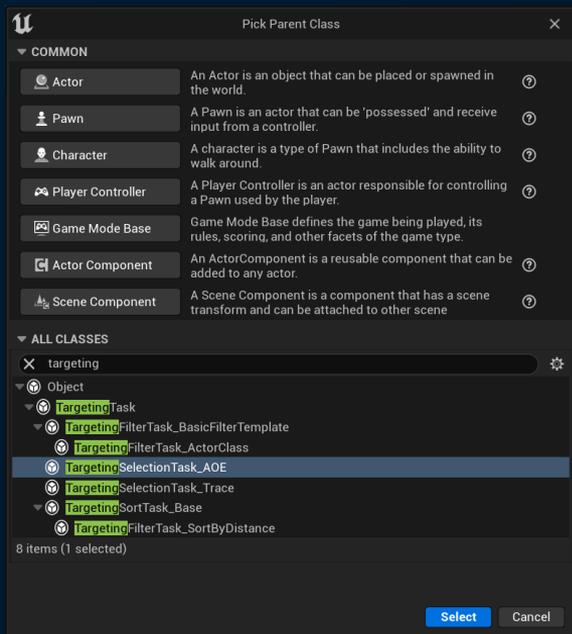
- **Targeting Task**を配列に登録 (先頭から評価)
- Targeting Selection ~ で  
まず対象のアクタをリストアップ
- フィルタやソートなどの加工処理を並べる



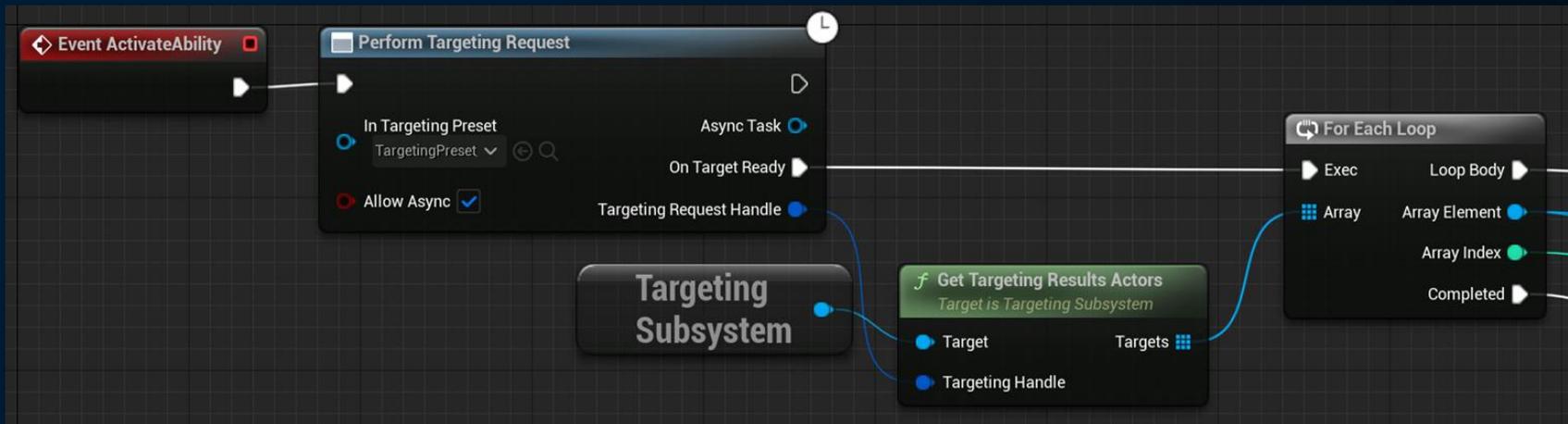
# C++またはブループリントでの拡張

- Targeting\*Task\* を継承してブループリントを作成

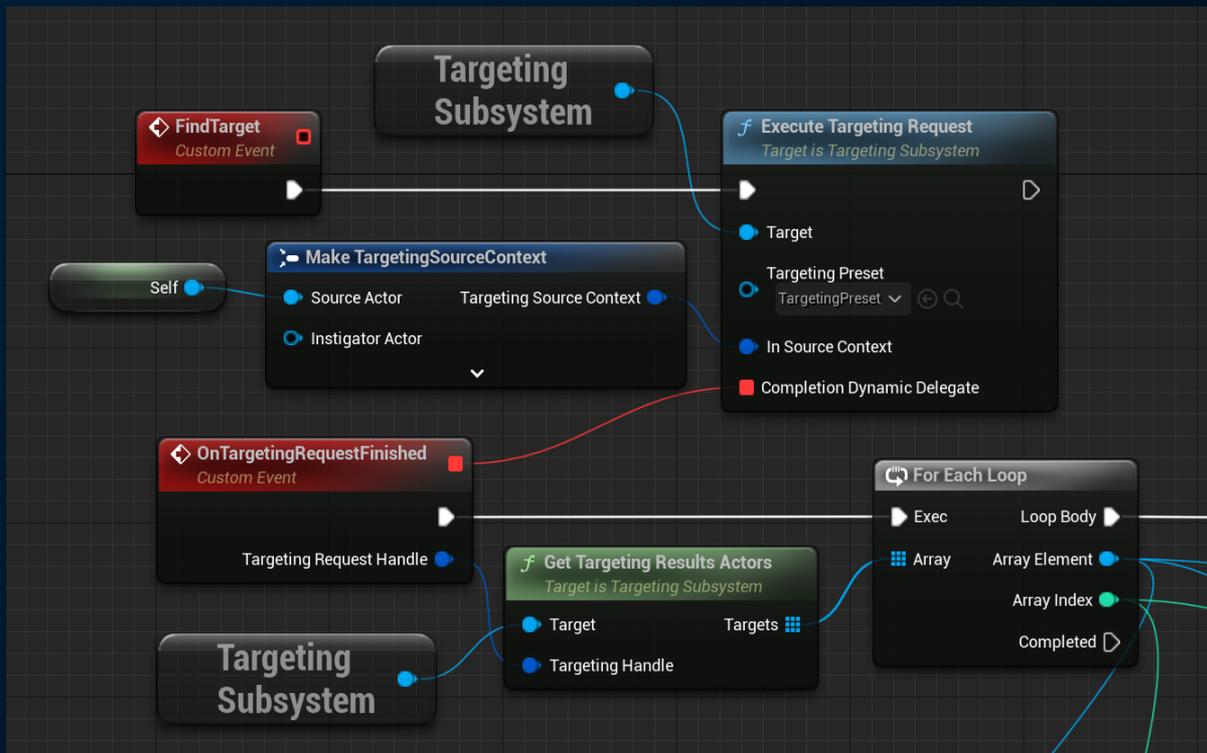
- UTargetTaskなどを継承したC++クラスを作成



# Targeting Task 実行 (GameplayAbilityから)



# Targeting Task 実行 GA以外から

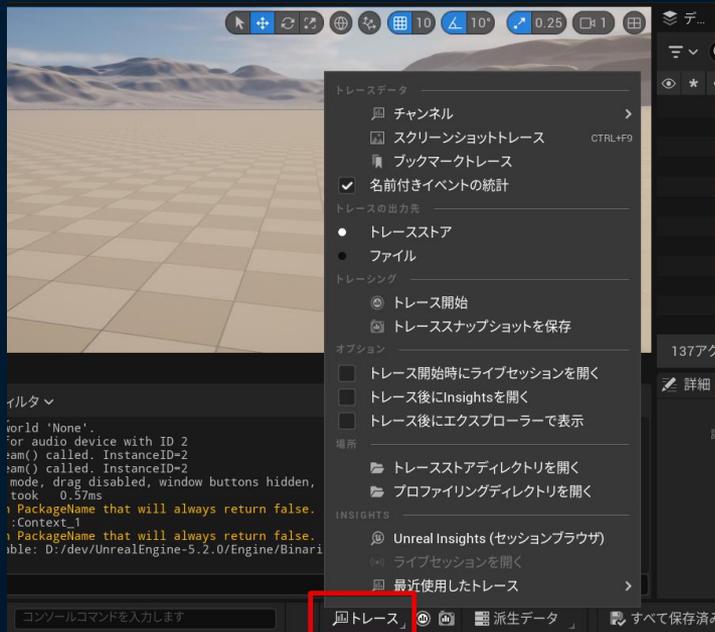




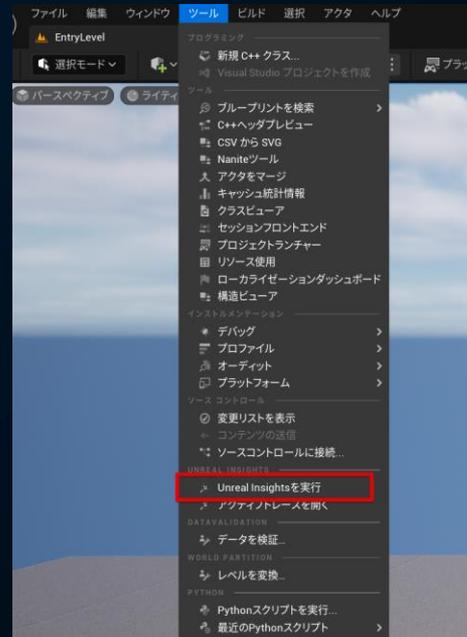
**UNREAL**  
ENGINE

**Unreal Insights 改善**

# エディターからの操作が大幅改善

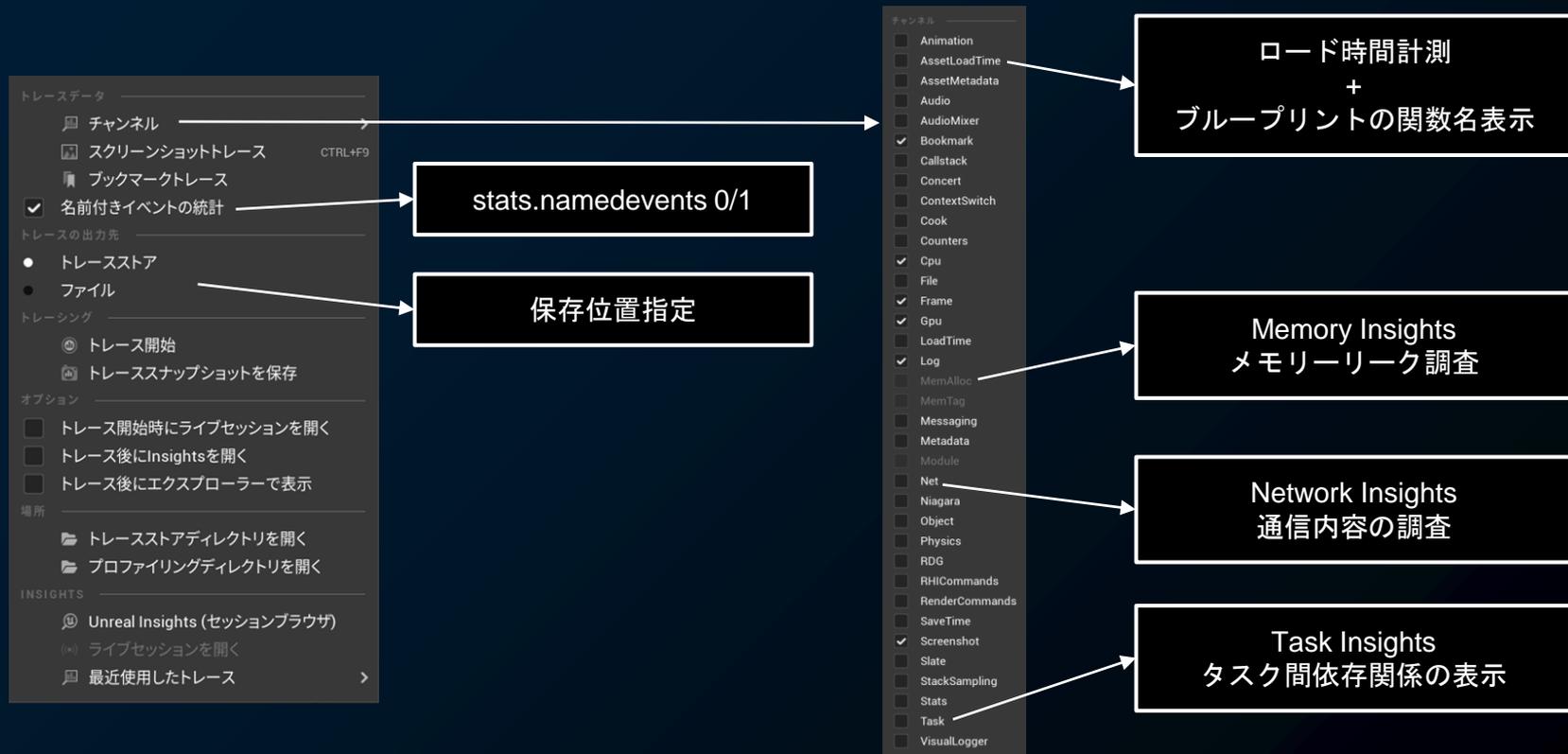


5.2 ~



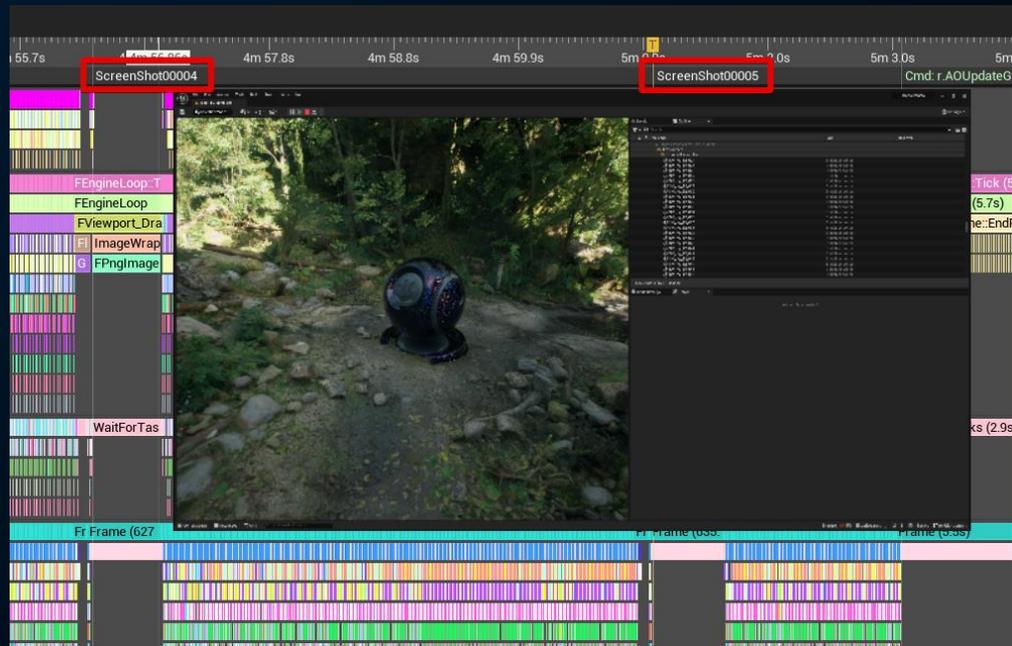
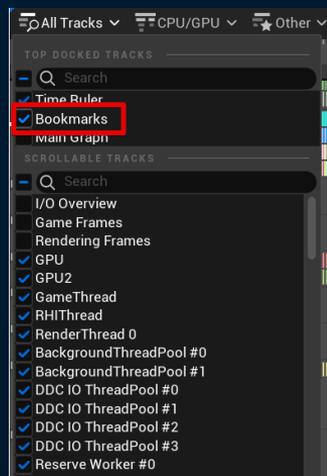
5.0 ~ 5.1

# トレースのメニュー



# スクリーンショットトレース

- スクリーンショットを記録
- Bookmark行でポイントして表示
- Trace.SnapshotSend

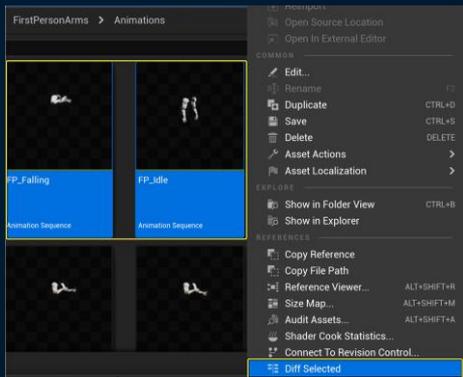




**UNREAL**  
ENGINE

# Diff/Review ツール

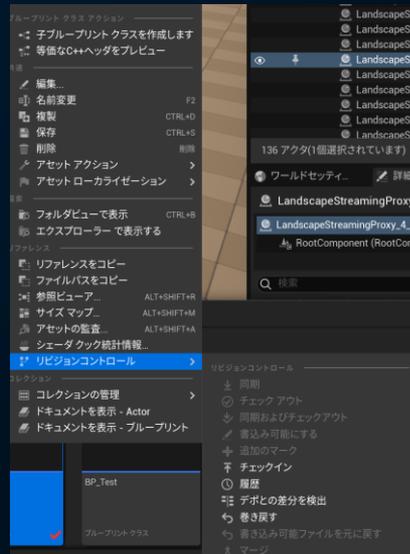
# Diffツール



二つのファイルの比較



ファイル履歴からの比較

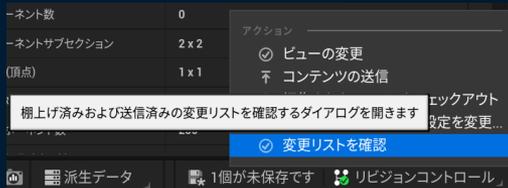


編集中のファイルとディポの比較

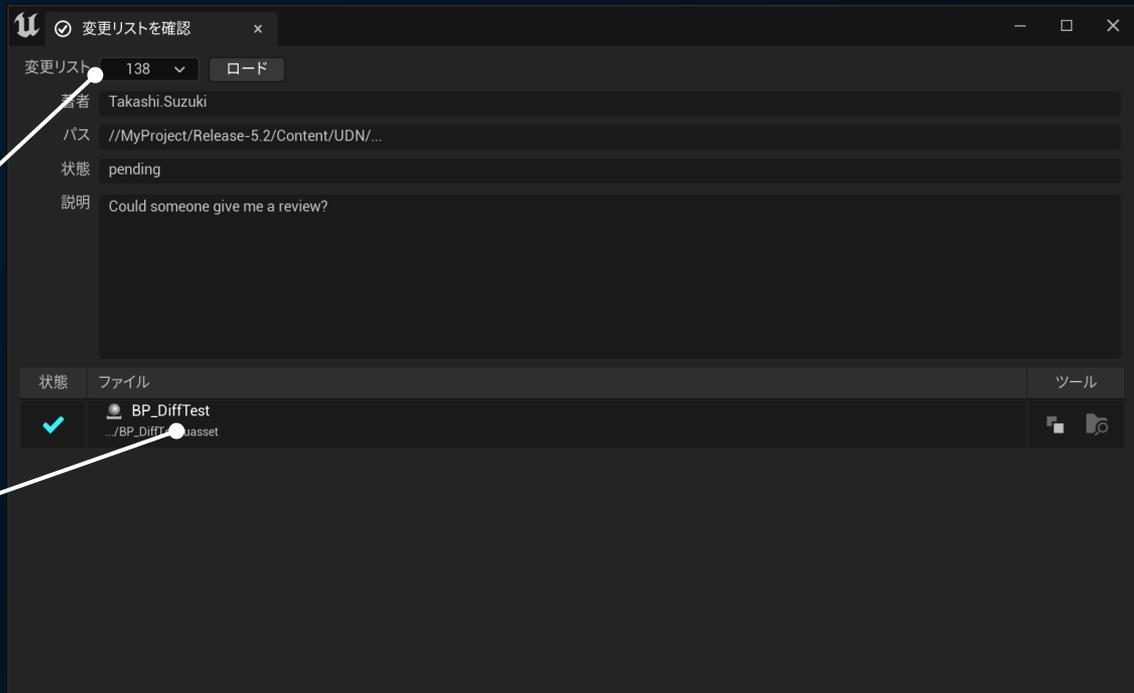
Unreal Diff ツール

<https://docs.unrealengine.com/5.2/ja/ue-diff-tool-in-unreal-engine/>

# Reviewツール



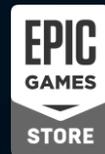
チェンジリスト番号を  
入力してロードを押す



ファイルを選んでDiff

# 最後に

- メールでのお問い合わせ先
  - [EGJ-support@epicgames.com](mailto:EGJ-support@epicgames.com)
  - [カスタムライセンシー様向け]
    - 技術的課題や問題点に関するご相談
    - パフォーマンスプロファイリング・チューニング
  - 弊社製品に関するご契約などのご相談



- 新たにライセンスをご検討される方
  - <https://www.unrealengine.com/ja/license#contact-us-form>
- 過去の講演のスライド | Docswell
  - <https://www.docswell.com/user/EpicGamesJapan>
- Unreal Fest 2023 Tokyo プレイリスト
  - [https://www.youtube.com/playlist?list=PLr\\_Cbd4sUDTzvdUTMe6cB5TVKDFZ8fMfC](https://www.youtube.com/playlist?list=PLr_Cbd4sUDTzvdUTMe6cB5TVKDFZ8fMfC)

