







# Unity Source Code Quick Guide - ダウンロードから内部デバッグまで



Partner Engineer



# Agenda

- → ソースコードについて
- → ビルドする方法
- → 実際にデバッグする
  - Windows Editor
  - Windows RUntime
  - Mac Editor
  - iOS
  - Android

→ デバッグ・調査のために書き換える

魚



## ソースコードについて

### ソースコードで出来る事?

**Unity**のRuntime/Editorをビルド可能なC++/C#コードにアクセスすることが出来ます。

- ・開示されていない処理の詳細について知ることが出来る
- → バージョンアップグレード時に発生する異なる挙動について調査可能
- -> ツール制作にも活かすことが可能
- → エラーログの文字列でGrepして、原因について調査可能
- → Profilerの重い箇所を文字列で該当処理の詳細が調査可能
- ・Unityそのものをデバッグする
- → Visual Studio / Xcodeで、BreakPointを利用したりしてデバッグする
- → 手元のプロジェクトデータからビルドされたものを使ってデバッグする事が出来ます。
- → ソースコードにログ処理やProfiler処理を追加しながらデバッグする

### ソースコードへのアクセス

- **Unity Enterprise**ライセンスが必要  $\rightarrow$ 
  - Enterpriseライセンスの後、個別の アクティベートが必要

- 通常はコンソール機のソースは アクセス不可
- Unity Dashボードよりアクセス可能  $\rightarrow$
- 利用はあくまで閲覧・デバッグ用  $\rightarrow$ 
  - 改変したUnityで製品リリースする には別途契約が必要
- → gitクライアントでアクセス可能

Ŷ	€ <sup>£⊋</sup> Unity Source Code				Unity Source Delive
	© ₽	Unity Source Code	ories Authentication		
@ +		Name	Platforms	Last activity	Ø
\$ * *		li2cpp-ps4-ps5-switch-xboxone-xboxx លើ	(IOS) ( macOS)     (MULInux)     (Mindows)     (Android) ( WebOL)     (Windows)     (Android) ( WebOL)     (Windows)     (W	Active 6 minutes ago	Code 🗸
		unity-ps4-ps5-switch-xboxone-xboxx ជ	(i) IOS (i) macOS (i) GNULINUX (ii) Windows (ii) Android) (iii) WebDL (ii) Playstation 4 (ii) Switch (ii) Switch (ii) Stoto One) (ii) Xbox Series X)	Active 6 days ago	Code 🗸

5

qitのインターフェース

- → Unityダッシュボード上でSSHキーを登録 する事でgitクライアントでのダウンロード が可能
  - ed25519のSSHキーが利用可能
- Unityダッシュボードからリンクされた形  $\rightarrow$ でGitのWeb UIインターフェースへアクセ ス可能
- → リリースされたバージョン毎にTagがある

🔯 탐색			<u></u> + -	£
및 unity-source-code/unit <>코드 ♡릴리즈 ☆활동	y-ps4-ps5-switch-xboxone-xboxx	비공개	④ 보기 16 및 포크	(
🕚 45 커밋	<b>양</b> 35 브랜치	[○ 1,102 태그	Search code	C
S 6000.0.40f1 * Go to file		<>Code +	설명	
Copybara Bot 307ba2a531	Code imported by Source control team	2주 전	Unity source code with support for Android, iOS, Linux, macOS, WebG	L &
fork	Code imported by Source control team	11개월 전	Windows. Additional platforms: Microsoft Xbox One, Microsoft Xbo	хс
sparse-checkout-rules	Code imported by Source control team	2개월 전	Series X, Nintendo Switch, Sony PlayStation 4, Sony PlayStation 5.	
Configuration	Code imported by Source control team	2주 전	🛱 Readme	
Documentation	Code imported by Source control team	2주 전	😫 341 GiB	
Editor	Code imported by Source control team	2주 전	Languages	
Extensions	Code imported by Source control team	9개월 전		
External	Code imported by Source control team	2주 전	● C# 56.3% ● C++ 35.7% ● C 2.1	%
Modules	Code imported by Source control team	2주 전	이 기타 2.8%	,
Packages	Code imported by Source control team	2주 전		
PlatformDependent	Code imported by Source control team	2주 전		
Platforms	Code imported by Source control team	지난달		
Projects	Code imported by Source control team	4개월 전		
Runtime	Code imported by Source control team	2주 전		
Shaders	Code imported by Source control team	6개월 전		
Tests	Code imported by Source control team	2주 전		

+ • 💮 • 약포크 0

Q

# ビルドする方法

### ビルドに必要なソフトウェア

### 共通

- git & git LFS(ダウンロード用) Windows
  - Visual Studio C++
  - Perl (Strawberry Perl 5.28+)
  - Windows SDK

### Mac OS

- Perl
- XCode
- mac OS SDK

(ソフトウェアのバージョンはビルドしたい Unityのバージョンによって異なる)

Unity version	Visual Studio	MSVC	Windows 10 SDK
6000.0	2022 17.9.3	14.39.33519	10.0.20348.0
2023.2	2022 17.4.2	14.34.31933	10.0.20348.0
2022.3	2022 17.4.2 + 2019 16.8.1	14.28.29333 (from VS2019)	10.0.20348.0
2022.2	2019 16.8.1	14.28.29333	10.0.20348.0
2022.1	2019 16.8.1	14.28.29333	10.0.20348.0
2021.3	2019 16.8.1	14.28.29333	10.0.18362.0
2021.2	2019 16.8.1	14.28.29333	10.0.18362.0
2021.1	2019 16.5.4	14.25	10.0.18362.0
2020.3	2019 16.5.4	14.25	10.0.18362.0
2020.2	2019 16.5.4	14.25	10.0.18362.0
2020.1	2019 16.2	14.22	10.0.17134.0
2019.4	2017 15.9	14.16	10.0.17134.0

#### リポジトリ Documentation/Inter

Documentation/InternalDocs/docs/BuildSystem/SetupWindows.md  $\downarrow \vartheta$ 

### まずはコマンドラインから

- → まずはコマンドラインから始まります
  - ドキュメントを閲覧するのもコマンド
    - Pythonが必要
  - ビルド用のコマンド
    - Visual Studio/Xcodeのプロジェクトファイルはビルドのコマンドから生成
    - ビルドもコマンドだけで出来る

### ドキュメント閲覧のコマンド

### 以下のコマンドを実行すると、ブラウザが立ち 上がりドキュメントを閲覧できます (Pythonが必要)

### Windows

- Documentation/InternalDoc/view.cmd Mac
  - sh Documentation/InternalDoc/view.sh

← C ① 127.0.0.1	8008/Gene	eral/basics.html	A® ☆ ☆ …
👙 Editor and Runt	Q Search		
			Table of contrasts
Guide	nent	Basic overview of Unity Editor and Runtime	
Get started	~	development	Git and GitHub for source
Introduction to URE development tooling		1	control
Branches and releases		A transfer	integration
First time developer setup		o important	Bokken for device provisioning
IDE configuration	>	This guide is pulled together from various internal docs sources, most of which are under development to some	Evergreen merge queue to
Work with the Unity repo	>	extent. The content is maintained collaboratively by Unity Editor, Runtime, and Ecosystem (URE) and Engineering	manage PRs to mainlines
Author your changes	>	Services (ES) teams. The markdown source is in the bocumentation/internations directory of the Unity repository. See the README there for guidance on contributing. Bug reports and content suggestions are welcome in devs-	
Build from source	>	internal-docs.	
Test your changes	>		
Work with Unity CI	>	This page provides a general overview of the key components of Unity Editor and Runtime	
Deliver your changes	>	development. For more detail about using these tools to deliver a change, see the Deliver your	
Unity knowledge base	>	changes.	

#### Our tooling

#### Git and GitHub for source control

The source code for the Unity Editor and Engine is stored in the unity/unity repository in a Unityspecific instance of GitHub Enterprise. Only Unity employees and contractors with appropriate permissions can access the repository.

#### Yamato for continuous integration

Yamato is our in-house CI system on which we run automated build and test workloads. Yamato is Git-native and jobs are defined in YAML. Most of the test configurations - sometimes referred to as "builders" for legacy reasons - are standardised, and developers are expected to run appropriate jobs on their PR to test it.

### ビルドコマンド

ビルドのコマンドは大きく二種類

- perl build.pl
  - 旧来の方式
  - プロンプトでビルドターゲットが選択可能
- jam コマンド
  - 基本はコチラを推奨
  - "jam WinEditor"のように引数でビルド対象を設定
  - 引数がわからない場合、引数なしで実行する事で候補が出力される
  - "jam WinEditor --help"のようにプラットフォーム+"--help"でプラットフォーム ごとの更なるオプションを表示

### 

## Visual Studio / Xcodeプロジェクトもビルドで生成

- jam ProjectFiles
  - VisualStudioやXcodeなどのプロジェクトファイル生成するコマンド
- Projects以下にVisualStudioやXcodeのプロジェクトが生成される
- 生成されたプロジェクトファイルを利用することで、デバッグが可能

## よく使うjamのビルドコマンド(1)

- jam WinEditor
  - WindowsのUnityEditorをビルド (Debugビルド)
- jam WinEditor -sCONFIG=release
  - WindowsのUnityEditorをビルド(Releaseビルド)
- jam MacEditor
  - MacのUnityEditorをビルド (x86)
- jam MacEditorArm64
  - MacのUnityEditorをビルド(Apple Silicon)

## よく使うjamのビルドコマンド(2)

- jam WinPlayer
  - Windows Standalone Player (Mono only)
- jam AndroidPlayer
  - Android Support (Mono only)
- jam AndroidPlayer64IL2CPP
  - Android Support( IL2CPP Arm8)
- jam iOSPlayer
  - iOS Support

**BREAKOUT SESSION** 

## ここまでの流れを ビデオで

## ソースコードへのアクセスとダウンロード



## 実際にデバッグする (Windows Editor)

UNITE/

## WindowsEditorデバッグ (デモ動画)

### 

# 実際にデバッグする (Windows Runtime)

WindowsRuntimeデバッグ (デモ動画)



# 実際にデバッグする (Mac Editor)

UNITE/

## MacEditor Debug (デモ動画)



# 実際にデバッグする (Android Runtime)

UNITE/

## Androidでのデバッグ (デモ動画)



# 実際にデバッグする (iOS Runtime)





**BREAKOUT SESSION** 

## デバッグ・調査のために 書き換える - Part1 -

UNITE/

書き換えのシチュエーション (1)

Profilerのキャプチャーフレーム上限を2000以上にしたい。

Preferences				_		×
Contraction Preferences						:
General 7 2D	Profiler					
Physics Al Navigation	Frame Count				-•	2000
Analysis Profiler	the Profiler Window through the 'Frame	an increase signi Count' setting.				sible in
Asset Pipeline	Show Stats for 'current frame'					
Colors	Default recording state		Enabled			•
Diagnostics	Default editor target mode on start		Play Mode			-
GI Cache	Custom connection ID					
Graphics Jobs	Connection ID cannot contain *, * or be to underscores.					
Package Manager	Target Frames Per Second (Highlights Module		60			
Scene Template						
Search						
Indexing						
Snader Graph Timeline						
UI Scaling						
Visual Scripting						

## Profilerのキャプチャーフレーム上限

ProfilerSettings.cs 🕫 🗙				
🕮 UnityEditor	<ul> <li>Control Control C</li></ul>			
24	public const stillig n_laigeti lamesi ei secolu - n_settiligsi i			
25   ; ;				
26	<pre>public const int kMinFrameCount = 300;</pre>			
27 💉	public const int kMaxFrameCount = 2000;			
28	<pre>public const int k_MinimumTargetFramesPerSecond = 1;</pre>			
29	<pre>public const int k_MaximumTargetFramesPerSecond = 1000;</pre>			
30				
31	private const int kMaxCustomIDLength = 26;			
32				
33	[SerializeField]			
34	private static int m FrameCount = 0:			
25				

"Modules/ProfilerEditor/Public/ProfilerSettings.cs"

のkMaxFrameCountの値を書き換えてビルド。

## Profilerのキャプチャーフレーム上限

Preferences				
		٩		
General ▲ ▼ 2D	Profiler			
Physics	Frame Count		• g	999
<ul> <li>Analysis</li> <li>Memory Profile</li> <li>Profiler</li> </ul>	Profiler overhead and memory usage in the Profiler Window through the	ge can increase signi 'Frame Count' settinç	ficantly the more frames are kept \ g.	visible
Asset Pipeline	Show Stats for 'current frame'			
Cinemachine	Default recording state		Enabled	
Colors	Default editor target mode on start		Play Mode	
Diagnostics	Custom connection ID			
External Tools GI Cache	Connection ID cannot contain ", * or converted to underscores.	r be more than 26 ch	aracters in length. Spaces will be	
Graphics	Target Frames Per Second (Highligi	hts Module)	60	
Jobs				
Packade Manade				

2000から 9999に変更してビルド。これで 9999フレームまでキャプチャ可能に

**BREAKOUT SESSION** 

## デバッグ・調査のために 書き換える - Part2 -

UNITE/

書き換えのシチュエーション(2)

### **ProjectSettings.SetDefaultShaderChunkCount**の効果を、 もっと詳しく知りたい

Chunkの展開処理や破棄処理をProfilerで可視化したい

Project Settings		— 🗆 X
🗘 Project Settings		
Adaptive Performa Audio Burst AOT Setting: Editor Graphics Input Manager Input System Pack Settings Memory Settings Package Manager Physics Settings	Player Shader Variant Loading Settings Default chunk size (MB)* Default chunk count* Override Scripting Define Symbols List is Empty	
Physics 2D Player Preset Manager Quality	Additional Compiler Arguments List is Empty	
Scene Template Script Execution O Services ShaderGraph Tags and Layers TextMesh Pro Time Timeline UI Toolkit Version Control Visual Scripting XR Plugin Manager	Suppress Common Warnings Allow 'unsafe' Code Use Deterministic Compilation <b>Optimization</b> Prebake Collision Meshes*	+ - Revert Apply

### **SetDefaultShaderChunkCount**を 簡単に説明

- ShaderBinaryDataはChunkに分割されてシリアライズ される
- それぞれのChunkはLz4圧縮されている
- Shaderがロードされたとき、全てのChunkは圧縮され た状態でメモリ上にロードされる
  - SetDefaultShaderChunkCountが0以外の時
- 必要に応じてLz4圧縮されたChunkを解凍して、メモリ
   上に展開される
  - ProfilerのShader.DynamicLoadGPUProgramに カウントされる
- ChunkCount数が指定された以上になると、展開後のメ モリは破棄される



Shader.DynamicLoadGPUProgram 0.187ms

Current frame accumulated time: 0.797ms for 2 instances on thread 'Worker 2' 1.27ms for 4 instances over 2 threads Shader: Hidden/VFX/VFXG\_Falling\_Sand\_Rocks/Small\_Rocks/I Pass: <Unnamed Pass 3> Stage: Keywords: FOG\_LINEAR INSTANCING\_ON \_FORWARD\_PLUS

## 今回、書き換えたい事はつまりこういう事

背景:

デフォルト状態では、"Shader.DynamicLoadGPUProgram"で、Chunk展開リクエストが発生したことがわかるが、具体的に何個目のChunkが展開されたかわからない。 Chunkの展開処理だけ知りたいChunkを捨てたかわからない。

今回は以下の二つの処理を追加する

- Chunk展開した処理をProfilerに載せる
- ・Chunkを捨てた処理もProfilerに載せる

Presentation



## Code example Profilerの処理の書き方@EngineCode(C++)

```
// PROFILER MAKERマクロを使って、ProfilerMakerをGlobal変数として宣言
 1
 2 PROFILER MARKER(
 3
         gShaderDynamicLoadGPUProgram, /* <- アクセスのためのGlobal変数 */
               kProfilerRender.
                                           /* <- カテゴリ */
 4
        "Shader.DynamicLoadGPUProgram", /* <- Profilerの名前 */
 5
        /* 以下がMetadataの型と名前を宣言 */
 6
        core::string, "Shader", core::string, "Pass",
        core::string, "Stage", core::string, "Keywords");
 8
 9
10
11
   // 実際に計測したい時は PROFILER AUTOマクロでScopeを自動的に計測
12 // ConstructorでBegin、DestructerでEndするようなインスタンスを自動で作ってくれる
13 void SubProgram::Compile(CompilationData& data){
   #if ENABLE PROFILER // 前に処理があるけど…省略
14
       // ここから、関数が終わるまでの間を自動で計測します
15
16
       PROFILER AUTO(gShaderCreateGPUProgram,
17
                   /* 以下メタデータ */
18
                   data.fromShader->GetScriptClassName(), passName, *stageDesc, variant);
19
   #endif
20 }
21
22
23
24
25
```

**BREAKOUT SESSION** 

## Let's Demo

UNITE/

## DynamicShaderLoadに詳細情報を追加するカスタム (デモ動画)

## まとめ

→ UnitySourceCodeはEnterpriseライ

センスのお客様はアクセス可能

- → 読む以外にも、デバッグを通して内 部処理の知識を深める事が出来る
- → 実プロジェクトデータを使ってエン ジンのデバッグも可能

欱

# Thank you

