摘要

[文档标题]

目录

[热更新流程 3](#_Toc903)

[1 构建流程 3](#_Toc13481)

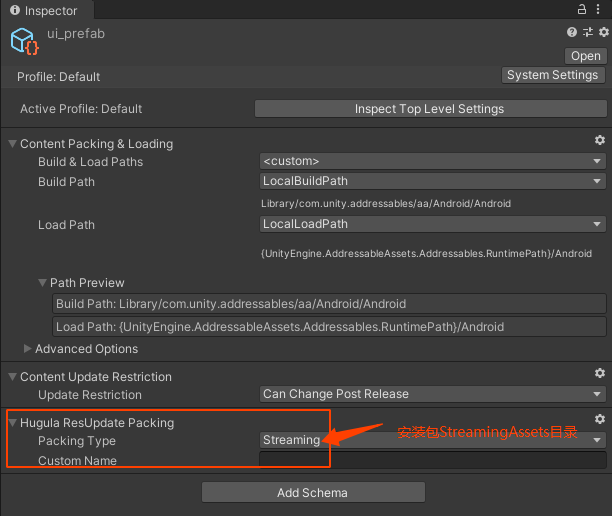
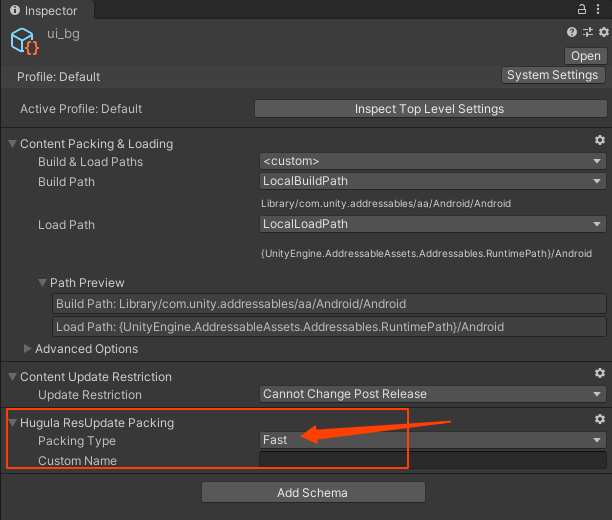
[2 发布流程 4](#_Toc4387)

[3 热更新流程 5](#_Toc7942)

# 热更新流程

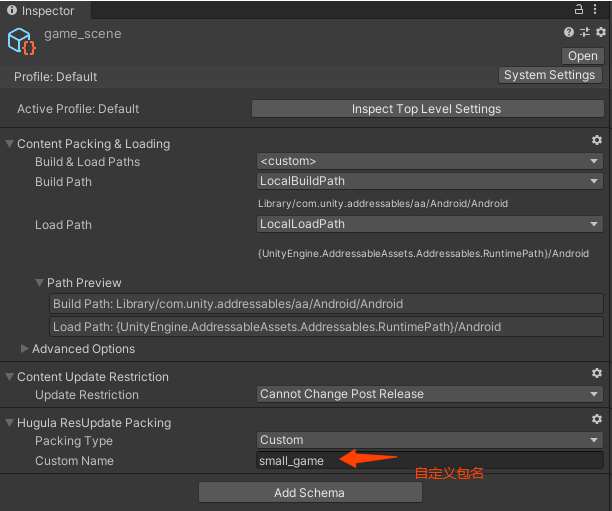
## 1 构建流程

1. 依赖AAS标准构建完成。
2. 根据HugulaResUpdatePacking配置信息生成FolderManifest文件列表信息。



HugulaResUpdatePacking中的PackingType有四种模式

* Streaming放入StreamingAssets目录，默认选项不会单独打包。
* Fast首次加载包，启动时候加载，可以配置是否等待加载完成进入游戏(可以支持android aab的fast模式)。
* Demand 按需加载，需要代码加载(类型aab的demand模式目前未支持)。
* Custom自定义包，根据需求自己定义包名(与demand类型可以自定义多个包)。



1. 处理Catelog文件，将Catelog信息放入streamingPack。
2. 清理缓存的热更新文件。
3. 构建自定义包的zip文件，根据上面生成的信息打包对应的bundle资源到zip包。
4. 构建本地文件列表，将foldermanifest打包成一个assetbundle(streaming\_all.u3d)并放入StreamingAssets目录作为本地版本对比依据。
5. 读取首包与本地文件列表，如果没有首包copy本地包作为首包。
6. 对比首包与本地包生成差异文件列表，增对每个文件夹生成差异文件列表。
7. 根据差异文件列表copy本地变更文件到热更新目录，生成变更文件列表作为远端对比依据。
8. 生成版本文件信息(version.json)。

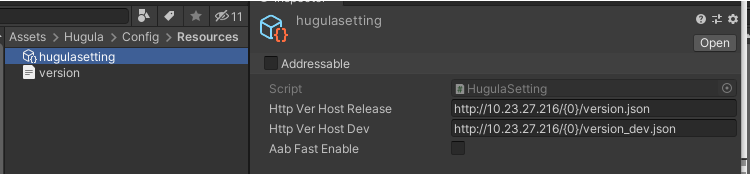
* 从Assets/Hugula/Config/Version中找到对应平台配置
* 生成本地对应版本配置覆盖当前版本
* 覆盖总版本配置
* 生成本地配置(仅仅用于 fast.zip ,demand.zip等自定义包下载)

1. PreprocessBuild流程（打包之前执行）

* 删除自定义包中的bundle文件。
* 如果开启了android aab模式，copy (fast)包的内容到工程PAD\_Build/custom\_fastfollow.androidpack中。

## 发布流程

1. 配置hugulasetting.asset



此处配置通过HUGULA\_RELEASE切换

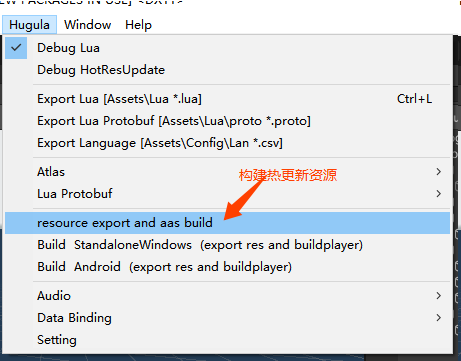
1. 多平台Assets/Hugula/Config/Version 信息配置

{platform}.json release配置

{platform}\_dev.json 开发模式配置

配置说明文档路径 Assets/Hugula/Config/Version/readme.txt

1. 构建热更新资源 （菜单Hugula/resource export and aas build）



1. 一键出包

Hugula/Build StandaloneWindows (export res and buildplayer) 构建pc版本并生成热更新资源

Hugula/Build Android (export res and buildplayer) 构建android版本并生成热更新资源

1. CDN资源上传

每次版本发布，将./UpdateRes目录下所有文件同步上传到远端服务器即可。

## 3 热更新流程

1. 读取本地streaming foldermanifest(streaming\_all.u3d)文件。
2. 读取可持续目录foldermanifest(streaming\_all.u3d)文件并合并到 streaming foldermanifest用于后面对比热更新资源。
3. 加载远端version.json 地址在Hugula/setting中http ver host配置, version.json内容配置位于Assets/Hugula/Config/Version目录。
4. 判断是否加载fast\_{crc}.zip 包。 利用远端version.json中的cdn\_host连接下载fast包。
5. fast\_{crc}.zip 包解压。 所有已经下载完成的zip包中的assetbundle重定向地址配置。
6. 对比远端版本号 判断下载远端foldermanifest(streaming\_all.u3d)文件。
7. 对比本地与远端foldermanifest(streaming\_all.u3d)文件下载变更的文件。
8. 下载完成，保存远端foldermaifest(streaming\_all.u3d)到可持续化目录，更新变更文件的assetbundle重定向地址配置。
9. 完成下载进入游戏。

## 4 BaCkGroundDownload.cs

///<summary>

/// 开始下载

/// </summary>

public void Begin()

///<summary>

/// 暂停下载

/// </summary>

public void Pause()

///<summary>

/// 下载zip文件

/// </summary>

public uint AddZipFolderManifest(FolderManifest folder, System.Action<LoadingEventArg> onProgress, System.Action<FolderManifestQueue, bool> onItemComplete, System.Action<FolderQueueGroup, bool> onAllComplete)

///<summary>

/// 下载多个zip文件

/// </summary>

public uint AddZipFolderManifests(List<FolderManifest> folders, System.Action<LoadingEventArg> onProgress, System.Action<FolderManifestQueue, bool> onItemComplete, System.Action<FolderQueueGroup, bool> onAllComplete)

///<summary>

/// 下载address依赖的zip包

/// </summary>

public int AddZipFolderByAddress(string address,System.Type type = null, System.Action<LoadingEventArg> onProgress = null, System.Action<FolderManifestQueue, bool> onItemComplete = null, System.Action<FolderQueueGroup, bool> onAllComplete = null)

///<summary>

/// 添加一组下载文件

/// </summary>

public uint AddFolderManifest(FolderManifest folder, System.Action<LoadingEventArg> onProgress, System.Action<FolderManifestQueue, bool> onItemComplete, System.Action<FolderQueueGroup, bool> onAllComplete)

///<summary>

/// 添加多组下载文件

/// </summary>

public uint AddFolderManifests(List<FolderManifest> folders, System.Action<LoadingEventArg> onProgress, System.Action<FolderManifestQueue, bool> onItemComplete, System.Action<FolderQueueGroup, bool> onAllComplete)

///<summary>

/// 重新加载失败的组

/// </summary>

public uint ReLoadErrorGroup(FolderQueueGroup group)

## 5 FileManifestManager.cs

/// <summary>

/// 本地最新数字版本号

/// </summary>

public static int localResNum

/// <summary>

/// 除fast包之外其他zip下载控制

/// </summary>

public static OtherZipMode otherZipMode

/// <summary>

/// 本地最新版本号

/// </summary>

public static string localVersion

/// <summary>

/// 检测bundle是否已经下载

/// </summary>

public static bool CheckBundleIsDown(string bundleName)

/// <summary>

/// 检查持久化目录的文件crc码

/// </summary>

public static bool CheckPersistentCrc(FileResInfo fInfo)

/// <summary>

/// check file is download from remote

/// </summary>

public static bool CheckIsUpdateFile(string abName)

/// <summary>

/// 通过address地址判断资源是否已经下载

/// </summary>

public static bool CheckAddressIsDown(string address, System.Type type = null)

///<summary>

/// 构建有依赖加载的资源address地址重定向

///</summary>

public static void GenOverrideAddressTransformFunc(string folderName, string defaultKey, System.Func<string, string> onTransform = null)

///<summary>

/// 清理有依赖加载的资源address地址重定向

///</summary>

public static void ClearOverrideAddressTransformFunc(string folderName)

///<summary>

/// 有依赖加载的资源address地址重定向

///</summary>

public static string OverrideAddressTransformFunc(string address)

自定义包下载示例demo/player\_load.lua

默认资源替换示例demo/welcome.lua 155-157 默认示例