**热更流程说明**

# 设计思想

1. 更新系统以资源包为单位进行更新
2. 更新流程中任意环节被中断，大退后都可持续正常运行更新系统
3. 动态读写文件因为读写时断电损坏，不应造成严重后果  
    严重后果包括但不限于：过多的资源需要重新更新；更新系统瘫痪

# 基本规则

## 不可热更的资源包

为了确保更新系统的健壮，以”rom\_”开头的资源包视为内核级资源包，不可热更。目前内核级资源包有以下两个：

**rom\_share** 存放更新系统和游戏内都需要用的资源，常驻内存不卸载

**rom\_upd** 存放仅被更新系统使用的资源，更新流程结束后从内存释放

# 宏含义

**<平台标识>** 区分不同的运行平台，例如ios android win64 ...

**<应用标识>** u3d的player Settings面板中 Bundle Identifier 属性

**<应用版本>**  u3d的player Settings面板中Version属性

**<PersistentData>** 外部可写文件夹，也可以认为是动态更新存放下载资源的根目录

**<包类型标识>** 包的类型，取值为pack\_script 或 pack\_res

# 资源更新流程

配置文件内容参见“配置文件”章节

## 应用大版本更新时清理外部资源包

needClean = true

versionPath = “<PersistentData>/lastclean\_app\_version.xml ”

If(FileExist(versionPath ))

{

exAppVersion = 从versionPath 中读出版本号

//外部存储的版本号和应用内部版本号一致

//表示应用版本未提升过，因此设需要清理标记为false

If(exAppVersion == <应用版本>)

needClean = false

}else

{

//有可能是因为闪退导致 app\_version.xml 丢失，因此仍然执行清理流程

needClean = true;

}

If(needClean)

{

删除文件 <PersistentData>/pack\_main\_version.xml

删除目录 <PersistentData>/pack\_script 中所有文件

删除目录 <PersistentData>/pack\_res 中所有文件

将<应用版本>写入 变量versionPath为路径的文件中

}

## 比对主版本号

**环境变量**

A = 从 “更新站点/pack\_main\_version.xml” 取得的xml文档

B = 从 “Resources/<平台标识>/pack\_main\_version.xml ”取得的xml文档

C= 从 “<PersistentData>/pack\_main\_version.xml ”取得的xml文档

D = 如果C存在且内容正确无误则取C,否则取B

**执行流程**

If(A.Identifier!=D.Identifier || A.Platform!=D.Platform)

更新出错 //有可能是更新人员手误，放错了文件

If(A.AppVer!=D.AppVer)//检查大版本更新

提示升级应用，并终止更新

If(A.PakVer<=D.PakVer)//检查资源包是否有变动

本地版本是最新的，不做任何变动

Else

执行下一流程

## 比对详细更新列表

**环境变量**

A = 从 “更新站点/pack\_version.zip” 取得的xml文档

B = 从 “Resources/<平台标识>/pack\_version.xml ”取得的xml文档

**执行流程**

If(A.Identifier!=D.Identifier || A.Platform!=D.Platform)

更新出错 //有可能是更新人员手误，放错了文件

Foreach(currInfo in A )//遍历A中记录的每一个文件

{

//在B中查找名字为currInfo.n的信息，如果不存在则返回null

localInfo = GetBInfo(currInfo.n)

Bool changed = false;

If(localInfo==null)//有可能是新增文件

changed = true;

else If(currInfo.v>localInfo.v)//有可能文件已升级

changed = true;

If(changed )//通过包内只读的版本信息比对结果是需要更新

{

//进一步确认文件是否已变更过

//每个文件都有一个对应的xml，记录它的版本号

//这个xml一开始是没有的，只有热更之后才会出现

ExVersionFile = “<PersistentData>/<包类型标识>/”+currInfo.n+”.xml”

If(FileExist(ExVersionFile)) //外部版本文件存在

{

exVersion = 从ExVersionFile 取出的版本号

If(currInfo.v<=exVersion)

{

//当远程版本号小于等于外部存储的版本号时，表示已经更新过，所以置状态为不更新

changed = false;

}

}

}//end If(changed )

if(changed )

{

下载更新包 更新站点/<平台标识>/<包类型标识>/currInfo.n+”.zip”

解压并存入路径 “<PersistentData>/<包类型标识>/”+currInfo.n

将远程该文件版本信息写入变量ExVersionFile表示的路径中

}

}//end Foreach(currInfo in A )

将从远程取得的资源包主版本号写入 <PersistentData>/pack\_main\_version.xml

资源更新完成

# 线上版发布流程

注：除红色高亮的步骤，其它的适用于平时研发版本发布，以及临时版本发布

## ResProject

1) 更新 ResProject

2) Tools/Configs Generate

3) Tools/Release Bsv (一骑当千2专用工具，其它项目也可自由扩展自己的工具集)

4) Tools/Resource Bundle Generate

## GameProject

1) 更新 ResProject

2) 如果是对外发布大版本，则<应用版本>需要增加1

3) Tools/Import Configs

4) Tools/Lua Pack Generate

5) Tools/Import Resource Bundle

6) Tools/Online Pack Generate

# 线上资源包更新流程

## 更新资源文件

web/pack\_res

web/pack\_script

## cdn推送资源文件

web/pack\_res

web/pack\_script

需等待一段时间，确保cdn所有节点都已经更新成最新版本

## 更新资源版本配置信息

web/pack\_version.zip

## CDN推送资源版本配置信息

web/pack\_version.zip

需等待一段时间，确保cdn所有节点都已经更新成最新版本

## 更新主版本信息

web/pack\_main\_version.xml

## CDN推送主版本信息

web/pack\_main\_version.xml

# 配置文件

## 工具生成，运行期只读的配置文件

### pack\_main\_version.xml

**功能**

记录资源包的主版本号。每当任意资源包有任意变更时主版本号都会被提升。因此，客户端在下载pack\_version.xml前，下载该文件和本地信息进行比对，如果本地版本号大于等于远程版本号，则无需更新。否则下载pack\_version.xml进行比对。

**存放位置**

Resources/<平台标识>/pack\_main\_version.xml (包内只读)

更新站点/pack\_main\_version.xml

**格式**

<xml Identifier="<应用标识>" Platform="<平台标识>" > <AppVer v="<应用版本号>" /> <PakVer v="<资源包版本号>" /> </xml>

**范例**

<xml Identifier="com.qikuai.yq2" Platform="win64" > <AppVer v="1.0" /> <PakVer v="1" /> </xml>

### pack\_version.xml

**功能**

记录所有资源包的版本号。用于比对某文件是否需要被更新

**存放位置**

Resources/<平台标识>/pack\_version.xml (包内只读)

更新站点/pack\_version.zip

**格式**

<xml Identifier="<应用标识>" Platform="<平台标识>">

<!--资源队列--!>

<pack\_res>

<a n="<资源名>" v="<版本号>" />

...

</pack\_res>

<!--脚本队列--!>

<pack\_script>

<a n="<资源名>" v="<版本号>" />

...

</pack\_script>

</xml>

**范例**

<xml Identifier="com.qikuai.yq2" Platform="win64">

<pack\_res>

<a n="aa001" v="1" />

...

</pack\_res>

<pack\_script>

<a n="pack\_script" v="1" />

...

</pack\_script>

</xml>

## 运行期动态生成的配置文件

### lastclean\_app\_version.xml

**功能**

记录上次清理外部资源包时的应用版本号，用于判定大版本提升，对外部资源包和外部存放的版本信息进行清理。

**存放位置**

<PersistentData>/lastclean\_app\_version.xml

**格式**

<xml> <version v="<应用版本号>" /> </xml>

**范例**

<xml> <version v="1.0" /> </xml>

### <包名>.xml

**功能**

记录某个具体资源包的版本号，该文件初始时不存在，只有当热更第一次后动态生成。

**存放位置**

<PersistentData>/<包类型标识>/<包名>.xml

**格式**

<xml> <version v="<包版本号>" /> </xml>

**范例**

<xml> <version v="2" /> </xml>