# Q&A

Zeus

//loginURL = "http://192.168.93.216:8080/login";

//loginURL = "http://192.168.93.253:8080/login"; //李永辉

Conn.cs

Gateway

Sess

## TAP

李永辉 yhli\_ex g3cz649U  
苗建超 jcmiao\_ex 9n2wl84D  
吴明生 mswu\_ex 362A8dtw  
  
登录地址： <http://tapd.tencent.com/>

## 宕机处理

try catch

## Shade

[Shader.WarmupAllShaders](http://www.baidu.com/link?url=MTt1e4HT6SkVPzdv7peBNhFSNwtNlkZr7AujVtgz3SNsCz2-m52BUicQdKhzjeVM)

## C#位运算

else if ((t & 0x8) != 0) {

yield return stream.read(4, tails);

ret = lead & 0x7UL;

ret |= (ulong)tails[0] << 3;

## kcp加密？,负载均衡服务器

string s = ConnHelper.formatBytes(buf);

## 长整数

ulongToNumber(id)

function tostring64(value)

local ret = nil

if type(value) == "number" then

ret = LuaNumber.doubleToString(value)

else

ret = tostring(value)

end

return ret

end

function equal64(value1, value2)

local v1 = tostring64(value1)

local v2 = tostring64(value2)

return v1 == v2

end

## Lua导出性能

原理分析：

http://blog.csdn.net/xingaaaxing/article/details/72869709

注册耗时，内存、查表，拆装箱

push 一个数字也不装箱

精确类型就没有

不要啥都用object传递和接收，那种出现装箱也只能怪自己

## 正则表达式查找替换

^.\*?([0-9]+\.).\*?href="(.\*?)".\*?>(.\*?数.\*?)<.\*?$

\1\3\n\2\n\n\n

^.\*public class (.\*) :.\*$

\1

**查找：**

**^.\*class (.\*) :.\*$**

**替换：**

**\_GT\(typeof\(\1\)\),**

## Redis

redis-server redis.windows.conf

## Go

[**liteide**](https://sourceforge.net/projects/liteide/)

Installing 13 tools at C:\Users\wumingsheng\go\bin

gocode

gopkgs

go-outline

go-symbols

guru

gorename

gomodifytags

impl

godef

goreturns

golint

gotests

dlv

Installing github.com/nsf/gocode SUCCEEDED

Installing github.com/uudashr/gopkgs/cmd/gopkgs SUCCEEDED

Installing github.com/ramya-rao-a/go-outline FAILED

Installing github.com/acroca/go-symbols FAILED

Installing golang.org/x/tools/cmd/guru FAILED

Installing golang.org/x/tools/cmd/gorename FAILED

Installing github.com/fatih/gomodifytags SUCCEEDED

Installing github.com/josharian/impl FAILED

Installing github.com/rogpeppe/godef SUCCEEDED

Installing sourcegraph.com/sqs/goreturns FAILED

Installing github.com/golang/lint/golint FAILED

Installing github.com/cweill/gotests/... FAILED

Installing github.com/derekparker/delve/cmd/dlv SUCCEEDED



D:\Program Files\Redis>"D:\Program Files\Redis\redis-server.exe" "D:\Program Files\Redis\redis.windows.conf"

set GOPATH=F:\project\zeus;F:\project\server

go install Login

go install Center

go install IDIPServer

go install Lobby

go install Match

go install Room

go install Gateway

Lobby.exe.infolog.2017.11.14.23

Center.exe

Gateway.exe

Room.exe

Match.exe

Lobby.exe

IDIPServer.exe

Login.exe

### 配置读取

github.com/spf13/viper

### 日志

<https://github.com/cihub/seelog>

### CPU

github.com/shirou/gopsutil/cpu

### 定义全局变量

var (

    // redis连接池

    pool \*redis.Pool

    // 给服务器间同步状态使用的redis连接池

    poolForServer \*redis.Pool

    // 单实例redis连接池

    poolForSingleton \*redis.Pool

)

## Events.fire怎么导给lua的

Lua中的Events.fire完全是独立的一套，跟C#完全没关系

## Svn

set GOPATH=E:\timefire\zeus;E:\timefire\server

go install Login

go install Center

go install IDIPServer

go install Lobby

go install Match

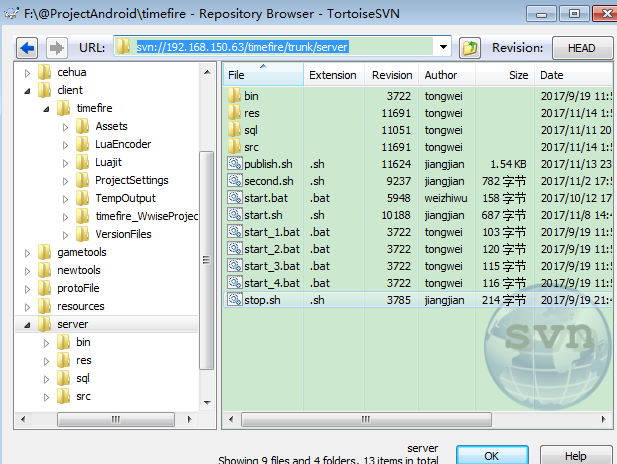
go install Room

go install Gateway

svn://192.168.150.63/mb/zeus/branch\_1

svn://192.168.150.63/timefire/trunk/server

svn://192.168.150.63/mb/zeus/branch\_1



## Terrain

## .asset

## 四元数

Quaternion shootRota = Quaternion.LookRotation(endPoint - startPoint);

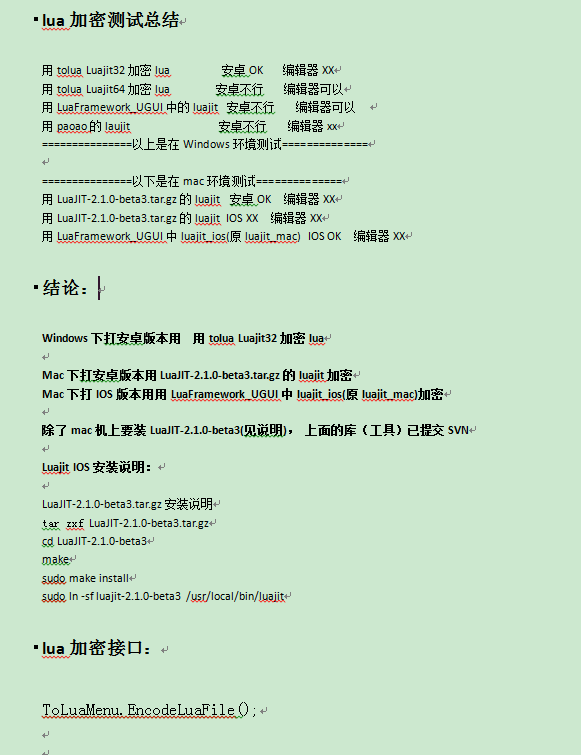
float yaw = shootRota.eulerAngles.y \* Mathf.Deg2Rad;

## Particle

particle.Stop();

particle.Play();

## Lua加密



## 安卓打包

安卓做测试包流程：

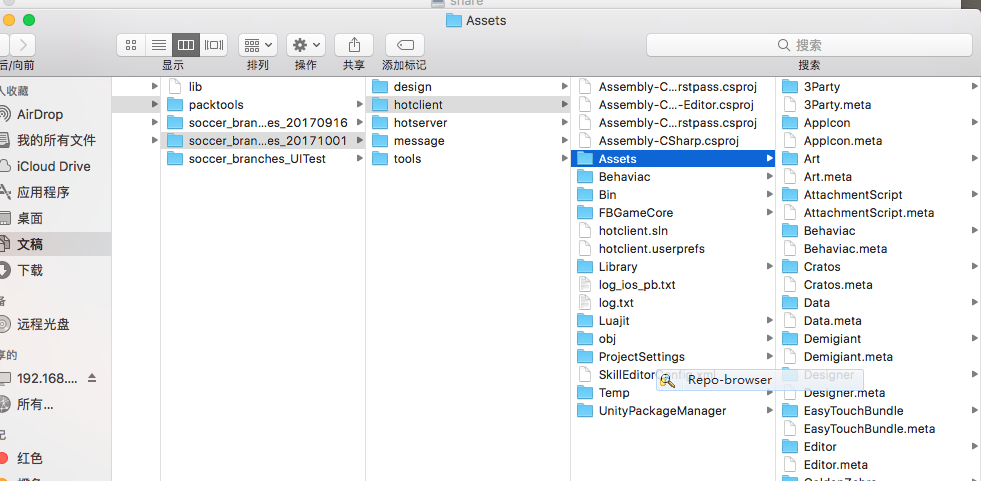
1、菜单Tools/一键打包安卓资源

2、BuildingSettings 打包

即可







## Mac mini

192.168.93.62

## 垃圾桶

192.168.96.231

soccer ztgame@123

## Luajit IOS环境

LuaJIT-2.1.0-beta3.tar.gz

tar zxf LuaJIT-2.1.0-beta3.tar.gz

cd LuaJIT-2.1.0-beta3

make

sudo make install

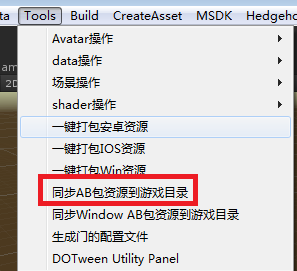
## 截屏

Ctrl+Shift+A

## lua

LuaProxy





## .asset

AssetDatabase.CreateAsset

## CG

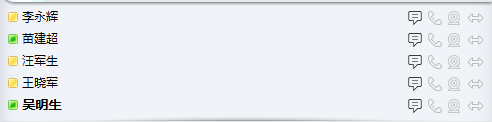
Handheld.PlayFullScreenMovie

## SVN

王晓军 14:13

svn://192.168.150.63/timefire/trunk/client

wumingsheng 12345678



## Git常用命令

### Push:

同步到服务器

git.exe push --progress "origin" Branch\_client20171019:Branch\_client20171019

### fetch:

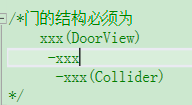
fetch只拉取不合并

git.exe fetch -v --progress "origin"

### Pull:

# Client

## 美术规范



## 功能

### 腾讯服务条款

**参考界面：**

CareerPanel

SettingsPanel

UI.close("LoadingPanel")🡪UIManager:close(panel)

**管理器UI.lua**

打开一个界面：

UI.show("LoginPanel") 🡪 UIManager.show()

UI.show("LoginNoticePanel")

每个UI预制都挂了一个UIProxy负责连接对应的luaCtrl和luaView

local enableAutoPickup = LuaPlayerPrefs.GetInt("AutoPickupItem", 0)

Toogle:

self.tg\_AutoPickTg:onStateChanged(function(isOn)

local value = isOn and 1 or 0

LuaPlayerPrefs.SetInt("AutoPickupItem", value)

Events.fire("onAutoPickupItemChanged")

end)

Toogle:

self.view.tgg\_TabBtns:addToggleEvent("Career",self.view.tg\_CareerInfo,function(isOn)

if isOn then

self.view.img\_CenterPanel:show()

CareerData:loadCareerData(2, ulongToNumber(AccountInfo.uid))

else

self.view.img\_CenterPanel:hide()

end

end)

self.view.tgg\_TabBtns:addToggleEvent("Friend",self.view.tg\_FriendRank,function(isOn)

if isOn then

self.view.img\_FriendPanel:show()

CareerData:loadFriendRank(2, ulongToNumber(AccountInfo.uid))

else

self.view.sr\_ScrollView0:clear()

self.view.img\_FriendPanel:hide()

end

end)

self.view.tgg\_TabBtns:setAllTogglesOff()

self.view.tgg\_TabBtns:switchTogglePage("Career")

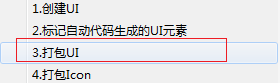
### UI

\*\*\*\*\*每个UI都有一个场景\*\*\*\*\*

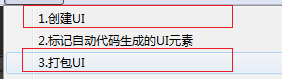
\*\*\*\*\* 通过场景编辑UI\*\*\*\*\*

Ctrl+S 自动生成和保存lua (程序用)

修改老的UI： 打开原有的场景，(策划美术用)



创建新的UI： 创建UI🡪打包UI ,(策划美术用)



\*\*\*\*要导出的控件

LoginPanelCtrl 登陆

NewMatchPanelCtrl 匹配界面

CameraPosionPanel 冒红？

MainPanel 战斗主界面

PackagePanel 背包

MapPanel 地图

EWNSPanel 方向刻度

LenXeightPanel 超级放大镜

LenXfourPanel 放大镜

LenHoloPanel 放大镜

LenRedPointPanel放大镜

SettingsPanel 设置

CustomBattlePanel 自动以操控？

### 登陆

Gateway.inst.login(accountName, passwd, data, loginURL, channel);

### 收发消息

客户端生成消息: newtools/protoGen/game.bat

服务端生成消息：protoFile/gen.py

打表：newtools/excelGen/genAll.bat

结构体消息：

ClientVertifyReq

**接收消息：**

(字节流)🡪(ID+字节流)🡪

RPCMsg消息只是一个特殊的结构体消息

Lua消息解析：

PARSE\_PROTO\_MSG

PARSE\_BIN\_MSG

PARSE\_HTTP\_MSG

onLuaEventParse

Lua消息发送：

function SEND\_PROTO\_MSG(msg)

local buffer = msg:SerializeToString()

local msgName = msg:GetTypeName()

Cratos.Zeus.inst:post(msgName, buffer)

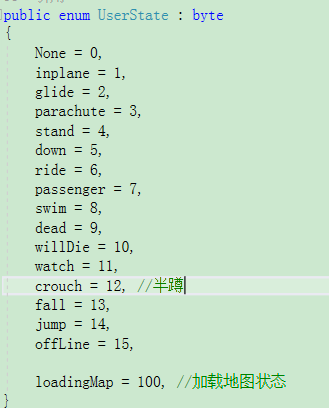
print("SEND\_PROTO\_MSG:" , "==protoMsgName==" , msgName)

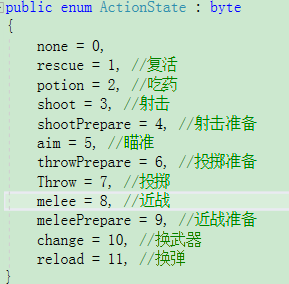
end

SEND\_RPC\_MSG

Cratos.Zeus.inst:rpc

### 状态





### OnEnterAOI

Space\_AOI

🡪GameScene\_Entity.oEnterAOI

🡪LuaBridge.onEnterAOI

🡪GameSceneData.lua OnEnterAOI

## 登陆

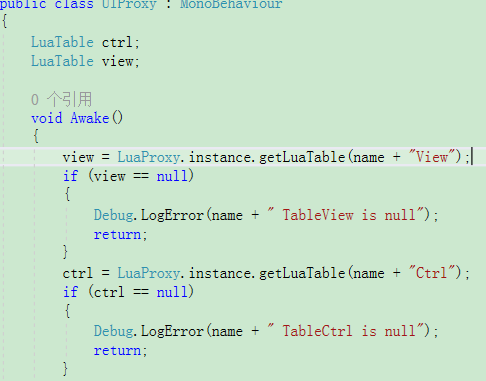
Debugger.LogError("the avatar is invalid", avatarid, userid)

## lua

用的tolua

Lua调用：

* main.lua
* 每个UI资源都挂了一个UIProxy， UIProxy负责关联对应的lua view和ctrl



## 初始化

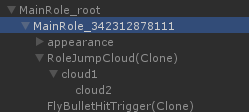
### lua

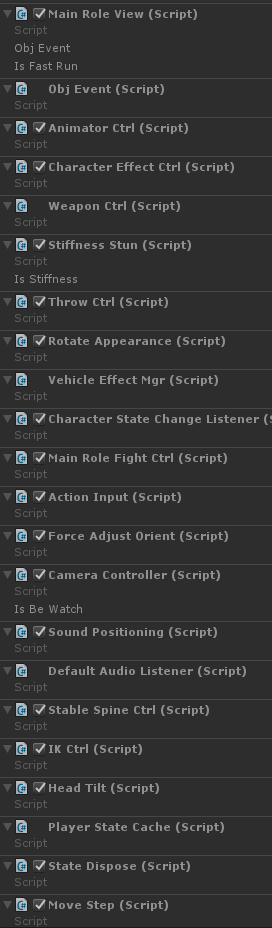
luaclient

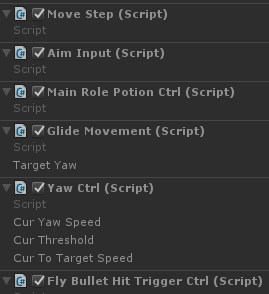
Game.Oninit🡪 LuaProxy.create()🡪LuaLoader(从luaclient继续)

## 动作

### 主角上挂的组件







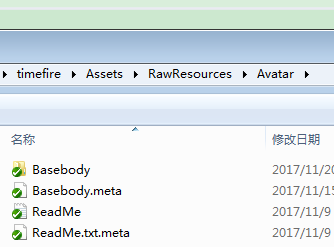
### 主角下面的RoleJumpCloud

下降过程中的云效果

### GlideCameraController

人物下降过程中上下震动

## 资源打包、管理、释放



整个basebody打成一个AB包？？

<https://www.uwa4d.com/u/overview.html>  
账号：ahsdwjs@163.com密码：wjs7632466

Resources.UnloadUnusedAssets();

**加载:**

Resources.Load

UnityEditor.AssetDatabase.LoadAssetAtPath

AssetBundleCreateRequestStatus

AssetBundle.LoadFromFileAsync(fullPath);

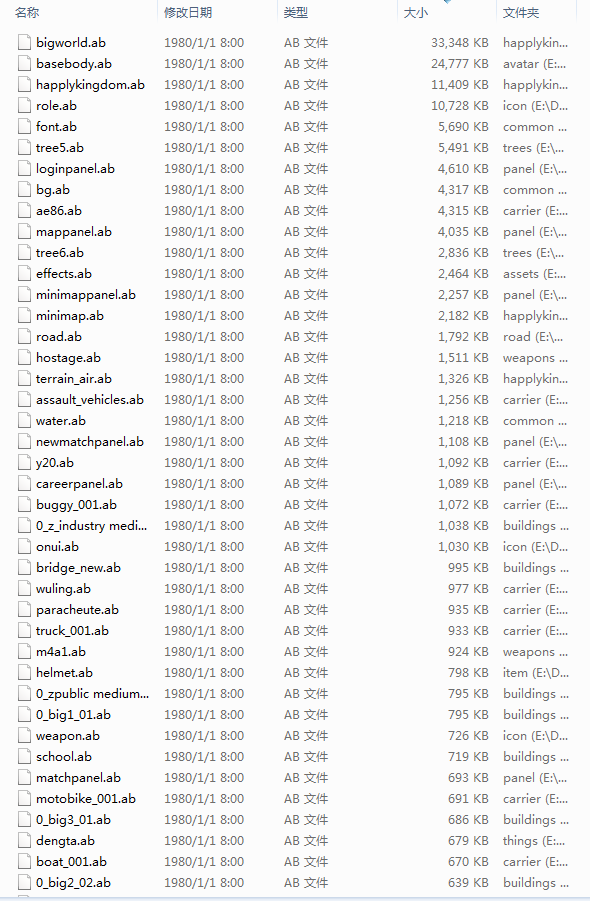
AssetBundleRequest req = loadedAB.LoadAssetAsync<T>(filePathInBundle);

while (!req.isDone)

{

yield return null;

}



## 场景

**异步加载(Unity接口)：**

SceneManager.LoadSceneAsync

**清除所有未使用资源(Unity接口)：**

Resources.UnloadUnusedAssets()

场景异步加载进度：

var sceneprocess = onSceneProcess(\_asycOperation);

### 主场景

GameScene\_Map

loadScene

🡪loadSceneAsync

🡪SceneManager.LoadSceneAsync(Unity)

编辑器下主场景(战斗场景)直接调用LoadSceneAsync加载

手机上先加载场景依赖的AB包，最后再通过LoadSceneAsync加载场景



社保中心 拉社保

## 网络

IOSIPv6

### 消息发送

**Buff链表:**



**消息结构示意图:**

**Sess.send(object msg) send(int msgID, byte[] buf)**

{

//二进制(结构体)、protobuf序列化

MsgCoder.inst.marshal(msg, msgBuf, out msgID, out msgLen))

}

🡺

**Sess.sendMsgBuf(int msgID , byte[] buf , int len)**

**{**

**//压缩加密包体**

connHelper.fillSendBuf

**TcpSocket.send**

**}**

🡺🡺

**TcpSocket.send(byte[] data, int offset, int length)**

{

找到buff链表的最后节点

if(最后节点有足够的空间)

拷贝字节流到链表最后节点

else

创建新的节点用来拷贝这些字节流(根据字节流的大小可能创建好几个节点)

从链表取出第一个buffer

调用TcpSocket.\_send(Buffer buffer)

}

🡺🡺

TcpSocket.\_send(Buffer buffer)

{

Try

{

异步发送一个buff (如果在回调检查中没问题，继续发送下一个buff)

}

Catch (Exception)

{

清空队列

释放内存

}

}

Lua:

function SEND\_PROTO\_MSG(msg)

Cratos.Zeus.inst:post(msgName, buffer)

Gateway.inst.send(msgInfo.msgID, msgBuf);

function SEND\_RPC\_MSG( ...)

Cratos.Zeus.inst:rpc(serverType, msgName, stream:GetUsedBytes());

Cratos.Zeus.inst:post

Gateway.inst.send(msgInfo.msgID, msgBuf);

### 消息接收

//每帧循环

TcpSock.update

🡺🡺

TcpSock.\_runRecvLoop

🡺🡺 IEnumerator sess.recvLoop (咋调用过来的)

2级🡺🡺

Sess.recvMsgBuf

3级🡺🡺

connHelper.readIO

{

**//rawIO**

TcpSock.recv(byte[] buffer, int offset, int length)

{

**从m\_receivedBuffers 获取字节流**

}

}

2级🡺🡺

MsgDeliver.inst.onMsgRecv(recvMsgID, recvBuf, recvLen);

{

**//反序列化（结构体消息、proto消息）**

MsgCoder.inst.unMarshal(msgID, msgBuf , msgLen , out msgName, out msg))

Events.fire(msgName, msg);

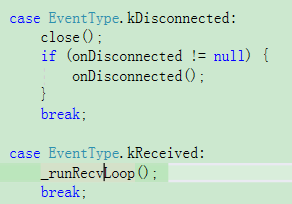
**//proto消息发到Lua层面一份（这只是放在队列里,真正传递见注2）**

rawMsg.put(new MsgContent(msgID, msgInfo.msgName, msgBuf));

}

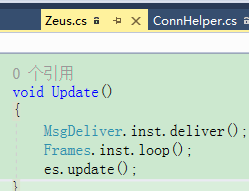
**注1：**

**有个独立的异步接收线程不断的接收数据放到 m\_receivedBuffers中，同时更新事件列表(**private LinkedList<EventType> m\_events**)操作数据上锁, 主线程根据事件链表进行操作：**



**注2：**

循环里不停取消息传给lua函数



**消息传给lua还有这三大接口:**



**=====================================**

### 套接字C#函数

m\_socket.**BeginReceive**(buf.data, 0, buf.data.Length, 0, \_onReceive, buf);

## 调试

Debug.DrawLine(startPoint, startPoint + shootRota \* Vector3.forward \* weapon.distance, Color.green, 10, true);

## 发布、更新

**宏：**

tx\_pb

PatchUrl.lua



PatchCfg.lua



//热更新完成之后调用

//销毁luaVM后重建

//清理c#缓存的lua数据结构

//重新加载lua的ab包资源

//重新进行lua层面的初始化

版本号：

version.cs

## 第三方库

### Unity插件

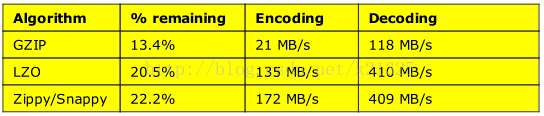
ShaderForge 、Haptic震动插件、EasyTouch 、BabeLua

### Ipv6

### LitJson

### Protobuf-net

### Snappy(压缩算法)



## 优化

### CPU

超过33ms的帧数占比 (?) (4.3%)

**a, 渲染模块**

**CPU消耗：**

红米Note2:

半透明(?) (1.5ms)

不透明(?) (1.8ms)

提升方法：

查看**Camera.Render**下各函数的消耗占比，对场景模型、蒙皮网格和UI进行控制，

要是Shader.SetPass消耗过高，建议对场景中的Material数量进行控制

**DrawCall:**

(?) (48--179)

**b, UI模块**

**CPU**

(?) (1.9ms)

**每帧堆内存**

(?) (2KB)

**函数：**

Canvas.SendWillRenderCanvases

**c, 动画模块**

**函数：**

Animator.Update (?) (1.8ms).

MeshSkinning.Update (?) (1.8ms)

**d, GC调用**

?帧/次 (1921帧/次)

1w帧 堆内存<30M

**检测方法：**

查看函数的堆内存

### 内存

(总内存峰值？ ) (249M)

(mono堆内存峰值 ?) (32.5M)

**a, Mono堆内存**

(?)(2.5W帧 < 40 M)

**a, 资源内存**

**纹理资源**

纹理资源数量峰值 (?) (618个)

纹理内存占用峰值 (?) (119M)

ETC1格式纹理个数 (?)

Alpha8格式纹理(?)

RGBA32(ARGB32)格式纹理：(?)

RGB24 格式纹理 (?)

RGBA16格式纹理 (?)

**Mesh资源**

资源数量 (?) (239个)

内存峰值 (?) (14M)

**AnimationClip资源**

资源数量 (?) (476个)

内存峰值 (?) (9.6M)

**AudoClip资源**

资源数量 (?) (101个)

内存峰值 (?) (7M)

### 耗电量

DrawCall

IO

流量

### 示例

A, 对频繁切换shade进行预加载

B, 减少GameObject的Active和Deactive次数 （会造成CPU浪费或间接造成CPU更大的浪费(比如Active出发Animator.Initialize)）

C, 场景合批

D，UI重新布局，降低DrawCall

E,2DUI转3DUI使支持动态合批

F,减少消息收发

# C#<-->lua接口

## 关联

LuaBridge

Luabind.lua

Events.fire("on\_RPCLuaEvent", "testplayermsg", newArgs);

## 已导出的

### 杂

\_GT(typeof(Screen)),

\_GT(typeof(HttpHelper)),

\_GT(typeof(Cratos.MsgInfo)),

\_GT(typeof(Cratos.MsgDef)),

\_GT(typeof(Cratos.Zeus)),

\_GT(typeof(LuaByteBuffer)),

\_GT(typeof(UserInput)),

\_GT(typeof(Game)),

\_GT(typeof(InputFilter)),

\_GT(typeof(DragDeltaChecker)),

LuaNumber

\_GT(typeof(WeaponCtrl.Weapon)),

\_GT(typeof(WeaponCtrl)),

\_GT(typeof(VehicleController)),

\_GT(typeof(AudioCtrl)),

\_GT(typeof(DragListener)),

\_GT(typeof(ClickListener)),

\_GT(typeof(StoppableDragDeltaChecker)),

\_GT(typeof(PressChecker)),

\_GT(typeof(ListElement)),

\_GT(typeof(ListView)),

\_GT(typeof(CycleListView)),

\_GT(typeof(AnimationCurve)),

\_GT(typeof(PressHoldButton)),

\_GT(typeof(ResourceLoader)),

\_GT(typeof(EffectResource)),

\_GT(typeof(UI.UIManager)),

\_GT(typeof(ModelResource)),

\_GT(typeof(AudioResource)),

\_GT(typeof(CommonResource)),

\_GT(typeof(UIResource)),

\_GT(typeof(ShaderResource)),

\_GT(typeof(SceneResource)),

\_GT(typeof(AvatarLoader)),

\_GT(typeof(ResourceManager)),

\_GT(typeof(GameScene)),

\_GT(typeof(RawImage)),

\_GT(typeof(GraphicQualityManager)),

\_GT(typeof(Rect)),

\_GT(typeof(MaskableGraphic)),

\_GT(typeof(Graphic)),

\_GT(typeof(RectTransform.Axis)),

\_GT(typeof(MovementJoystick)),

\_GT(typeof(Input)),

\_GT(typeof(UnityEngine.EventSystems.PointerEventData)),

\_GT(typeof(UIGallery)),

\_GT(typeof(UIGalleryItem)),

\_GT(typeof(UIRoleItemContent)),

\_GT(typeof(T\_Object)),

\_GT(typeof(EasyTouch)),

\_GT(typeof(Gesture)),

\_GT(typeof(Shader))

### C#、lua相互通讯

LuaBridge

### UI

\_GT(typeof(UIHelper)),

### 工具、扩展

\_GT(typeof(Debugger)).SetNameSpace(null),

\_GT(typeof(Extension)),

### 热更相关

//Custom Patch

\_GT(typeof(Patches.PatchChecker)),

\_GT(typeof(Patches.Patcher)),

\_GT(typeof(Patches.DownLoader)),

\_GT(typeof(Patches.UserDefault)),

\_GT(typeof(Patches.FileUtil)),

\_GT(typeof(Patches.MD5Code)),

### SDK相关

\_GT(typeof(TXSdkManager)),

\_GT(typeof(Msdk.LoginRet)),

\_GT(typeof(Msdk.WakeupRet)),

\_GT(typeof(Msdk.RelationRet)),

\_GT(typeof(Msdk.ShareRet)),

\_GT(typeof(Msdk.NoticeInfo)),

\_GT(typeof(Msdk.NoticeInfoList)),

\_GT(typeof(Msdk.PersonInfo)),

\_GT(typeof(Msdk.ePlatform)),

\_GT(typeof(Msdk.eQQScene)),

\_GT(typeof(Msdk.eStatusType)),

\_GT(typeof(Msdk.eWechatScene)),

\_GT(typeof(Msdk.TypeInfoImage)),

\_GT(typeof(Msdk.TypeInfoVideo)),

\_GT(typeof(Msdk.TypeInfoLink)),

\_GT(typeof(Msdk.TypeInfoText)),

\_GT(typeof(Msdk.ButtonApp)),

\_GT(typeof(Msdk.ButtonWebview)),

\_GT(typeof(Msdk.ButtonRankView)),

\_GT(typeof(Msdk.eMSDK\_SCREENDIR)),

\_GT(typeof(Msdk.eADType)),

\_GT(typeof(Msdk.eApiName)),

\_GT(typeof(Msdk.LocalMessageAndroid)),

\_GT(typeof(Msdk.LocalMessageIOS)),

\_GT(typeof(Msdk.GroupRet)),

\_GT(typeof(Msdk.ADRet)),

\_GT(typeof(Msdk.CardRet)),

\_GT(typeof(Msdk.WebviewRet)),

\_GT(typeof(Msdk.RealNameAuthRet)),

\_GT(typeof(ShareAssist)),

\_GT(typeof(TXSDKHelp)),

\_GT(typeof(RadarGraph)),

### 未导出的

### 还要导出的

Events（注册了没找到？）？ Frames？

Entity？视图相关的

AudioManager

DynamicGameObjectManager

读表?

BulletsMgr?

加入新的动作状态如何热更？

DoorManager

语音(? 不用。。)

**目标是把角色状态动作都导给lua**

# Todo

加密lua

C#全部接口导出给lua

AB资源规划

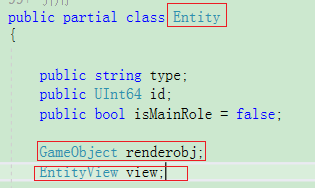
游戏优化

# 特点

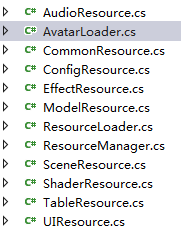
## 自动和服务器同步属性 （Entity\_Prop(文件)）

## RPC

## 逻辑视图分开



## 各种ResourceLoad



## Lua代码与UI预制名字一一对应，非常好找

## 按游戏状态划分UI

# 待改进？

C#调用lua太频繁，易产生GC

MonoBehaviour用得太多。。。