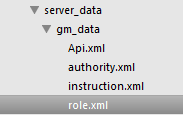
# Server

## GM指令

instruction.xml

GM\_Func.lua



@su admin

@add\_diamond 100

帮别人充值指令：

@gm\_charge\_ex user\_test(账号ID) 28(角色ID) 5(RMB)

调试

@gm\_mail 1 2 3 4 5

@gm\_mail whoever title content id1=num1,id2=num2,id3=num3 27(数据库id)

## DB

logdb.sql

# Client

## 特点

### 事件触发回调

### 配置全是xml

### RPC

* 自动注册Define文件里的网络回调函数
* 自动根据网络数据设置实体属性值
* 主角的RPC回调按功能分散在各个文件里(MySelfMail、MySelfFriend…)

// public partial class EntityMyself

### 通用的继承基类

GameData🡸 ItemParentData🡸 ItemEquipmentData

### 大量回调

### AB包通过WWW加载

### 界面逻辑与显示分离

### 行为树、状态机

### 主角动画控制位移

### Monster用了NavMeshPath

### 移动跟服务器没有交互

## 流程

初始化流程：

LoggerHelper.Debug("@@@ReleaseMode-->false");

SystemConfig.LoadServerList();

VersionManager.Instance.Init();

VersionManager.Instance.LoadLocalVersion();

gameObject.AddComponent("MogoInitialize");

IsRunOnAndroid = false;

gameObject.AddComponent("PlatformSdkManager")

## UI

主界面：

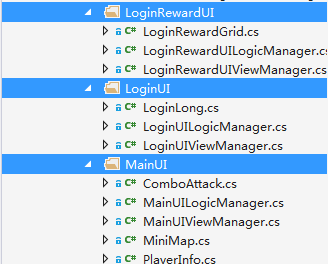
**MogoMainUI**

登陆前的登录界面：

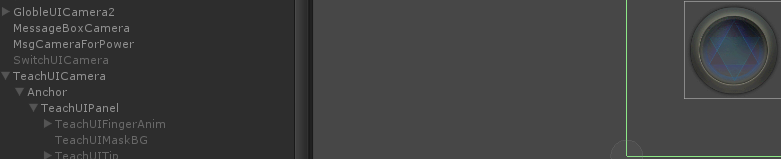
LoginUI.prefab

登陆后的登录界面：

NewLoginUI.prefab



### 摇杆：



## dll

### Common

EntityDefProperties 负责实体属性解析

EntityDefMethod 负责实体方法解析

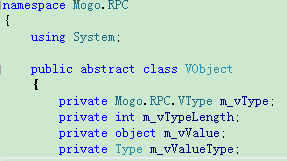
VObject

EntityDef \*\*\*\*\*解析好的实体放在这

DefParser \*\*\*\*\*负责解析

**类型定义基类：**

VObject



**RPC:**

### GameData

### LoaderLib

### MsgPack

## 用到的反射

1

this.GetType().**GetMethod**

2

var prop = type.**GetProperty**(propInfo.Name);

try

{

if (prop != null)

{

// Mogo.Util.LoggerHelper.Debug("prop: " + prop.Name + " value: " + value);

prop.**SetValue**(this, value, null);

3

Utils.ParseLuaTable(\_info, typeof(ItemInstance), out obj);

4

FormatXMLData

PropertyInfo[] properties = type.**GetProperties**();

5

FieldInfo field = type.**GetField**("fileName");

**6根据类型构造默认的对象？**

object obj3 = type.GetConstructor(Type.EmptyTypes).Invoke(null);

7 调用类型里的函数

dicType.GetMethod("Add").Invoke(obj2, new object[] { pair.Key, obj3 });

## 路径

public static string **AndroidPath**

{

get

{

return (Application.persistentDataPath + "/MogoResources/");

}

}

**OutterPath：**

安卓平台=AndroidPath

**m\_resourcePath：**

if (this.m\_isUseOutterConfig)

{

this.m\_resourcePath = SystemConfig.**OutterPath** + "data/";

this.m\_fileExtention = ".xml";

}

else

{

this.m\_resourcePath = "data/";

this.m\_fileExtention = SystemConfig.CONFIG\_FILE\_EXTENSION;

}

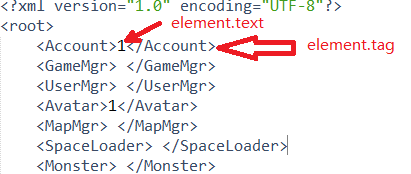
## 配置

所有配置采用xml? Mono.Xml

泛型编程 模板函数 反射？ 自动加载

### GlobalData.xml

### entities.xml



1表示需要校验MD5码

### resourceInfo.xml

各种资源的路径

### cfg.xml

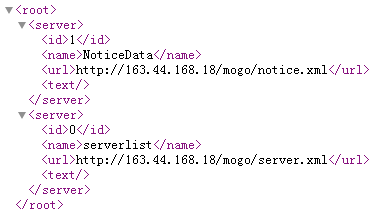
cfg**PATH:**

public static readonly string CfgPath = (Application.persistentDataPath + "/" + "cfg.xml");

🡺

<http://163.44.168.18/mogo/cfg.xml>

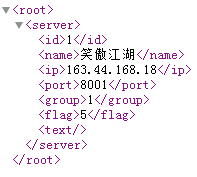
🡺



=====================================================================

<http://163.44.168.18/mogo/server.xml>

🡺



### server.xml

<?xml version="1.0">

<root>

<server>

<id>1</id>

<name>Port 8001</name>

<ip>163.44.168.18</ip>

<port>8001</port>

<group>1</group>

<flag>5</flag>

<text/>

</server>

<server>

<id>2</id>

<name>Port 8002</name>

<ip>163.44.168.18</ip>

<port>8002</port>

<group>1</group>

<flag>2</flag>

<text/>

</server>

### SystemSwitch.xml

<root>

<ReleaseMode>0</ReleaseMode>

<DestroyResource>0</DestroyResource>

<UseFileSystem>0</UseFileSystem>

<UseHmf>0</UseHmf>

<DestroyAllUI>0</DestroyAllUI>

<UsePlatformSDK>0</UsePlatformSDK>

</root>

### ItemEquipment.xml

装备道具表

## todo

UI调用