

 博客

登录 | 注册

Q

≡

燕良@游戏开发

--做游戏是一件快乐的事



目录视图

摘要视图

RSS 订阅

个人资料



房燕良

+ 加关注

发私信



访问: 125161次

积分: 1743

等级: 

排名: 第12304名

原创: 27篇 转载: 0篇

译文: 1篇 评论: 160条

引擎技术交流QQ群



游戏引擎能吃吗

(264656505)

文章搜索

Q

常用网址

Unity3D官网

Unreal Engine官网

游戏蛮牛

文章分类

Unity3D (19)

Unreal Engine (2)

自研引擎 (2)

其它 (5)

CSDN专家精选，微信开发学习路线大有看头！ 【博乐】点评美文，得C币 iOS开发前沿与Swift探秘 Swift初级教程大汇总

原

《炉石传说》架构设计赏析(1)：游戏启动流程

分类: Unity3D

2014-09-08 22:43

13895人阅读

评论(14)

收藏

举报

unity3d

游戏

c#

暴雪

炉石传说

前些天看新闻，Unity Awards两项大奖颁给了暴雪的《炉石传说》，这真是对Unity一个再好不过的宣传了——你看，暴雪都开始用Unity了。大家都知道，目前Unity发布的游戏大多都没有对程序集进行混淆、加密，所以作为一个炉石的玩家&Unity爱好者，自然不能错过这个机会。让我们好好看一下暴雪的代码吧。

炉石传说的游戏内容的非加密性，让我花了一些时间分析了其程序集，将一些设计思路记录下来，与大家分享。欢迎各路高手拍砖。

注明出处：[燕良@游戏开发](http://blog.csdn.net/neil3d/article/details/39104329)，<http://blog.csdn.net/neil3d/article/details/39104329>。



做这些分析的主要目的是

快速回复

- 看看炉石如何组织游戏逻辑，以支撑复杂的技能逻辑、表现等；
- 看看炉石是如何使用Unity的，其结构设计和技巧上有什么值得学习的地方；
- 向暴雪的程序员好好学习一下英语。

下面我们就正式开始。我习惯先分析一下游戏的启动流程，这中间就涉及到了游戏基础数据的管理&初始化，各种管理器级别的类，以及相互引用关系也会初步显现。

首先看一下我找到的一些游戏启动过程相关的类，下面是他们的类图：

文章存档

2015年08月 (1)

2015年07月 (3)

2015年04月 (1)

2015年03月 (1)

2015年02月 (1)

展开

阅读排行

《炉石传说》架构设计赏

(13888)

你不再需要TinyXML，推

(11247)

使用Unity3D的50个技巧

(10071)

分享一个小工具：Excel

(7735)

2009年混合语言编程总结

(7620)

《炉石传说》架构设计赏

(7036)

Unity3D-RPG项目实战 (

(4863)

Unity 4.6的使用匿名dele

(4715)

《炉石传说》架构设计赏

(4589)

Unity3D-RPG项目实战 (

(4539)

最新评论

次世代关卡制作流程：使用Unre

风吹夏天：大神回来了

分享一个小工具：Excel表快速转

valiant303：解压看到一个

excel2json.exe的可执行文件，

但是没看到你提到的例子表格，

我该怎么转换自己...

《炉石传说》架构设计赏析(2)：！

ZeroDeep：QQ群多少啊

分享一个小工具：Excel表快速转

胖小了个花：感谢分享了哈，随便

试了下可以的。不过我要组织好

excel的格式。我想问下楼主，如果

我的excel里...

分享一个小工具：Excel表快速转

房燕良：@liyingwill123:已经上

传：

http://download.csdn.net/detail...

分享一个小工具：Excel表快速转

liyingwill123：分享个编译的版本

吧，不可能为了用这个我还要装

个visual Studio啊

次世代关卡制作流程：使用Unre

王静娜：学习了，给个赞

次世代关卡制作流程：使用Unre

jump fire：好厉害的感觉。

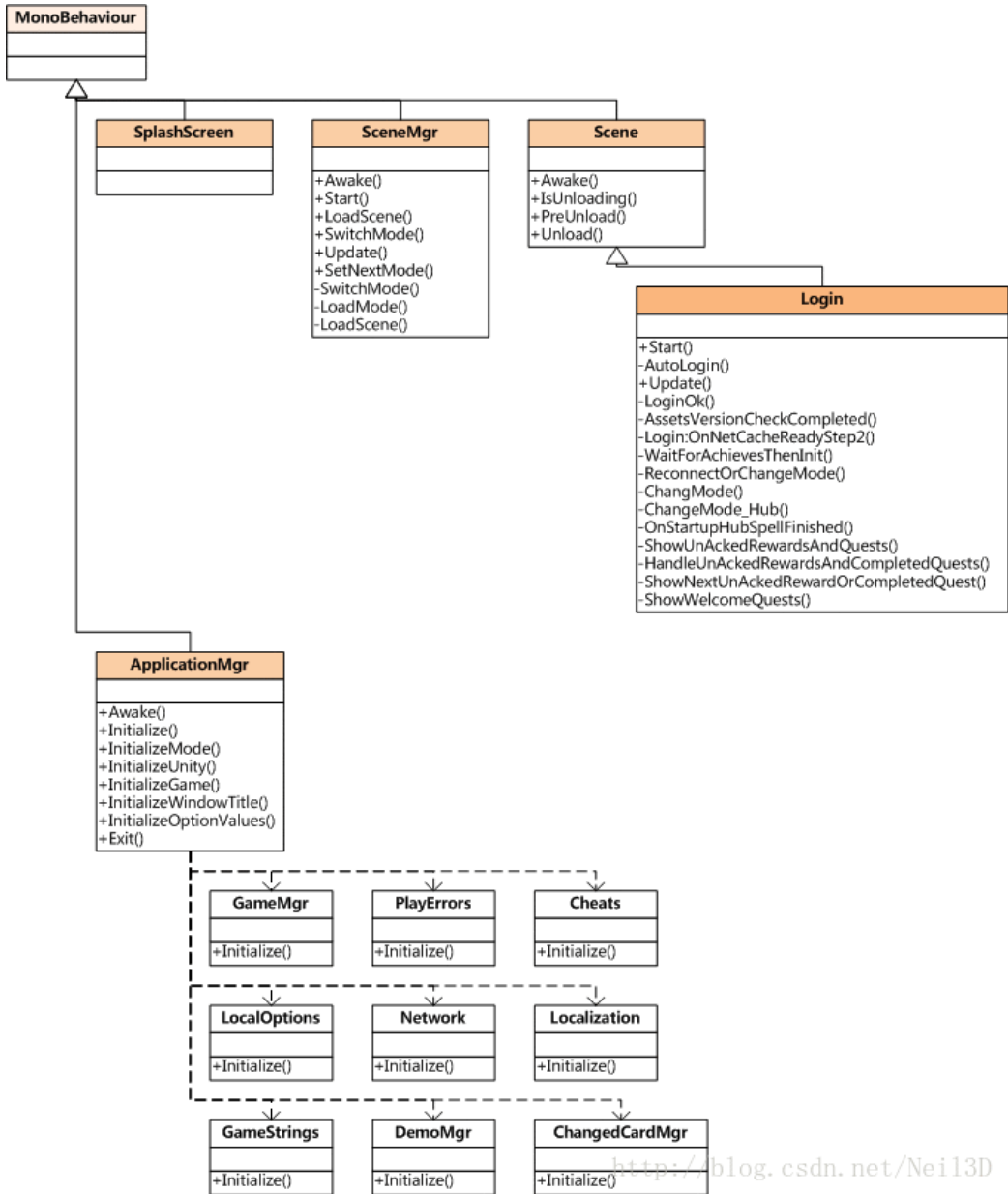
(๑*۞*๑)

次世代关卡制作流程：使用Unre

Json-Niu：好高大上，学习了。

次世代关卡制作流程：使用Unre

717606641：学习了，感谢博主



接下来我们分析一下游戏启动的操作流程。

- ApplicationMgr对象应该绑定到了一个场景对象中，这个场景应该在游戏启动中加载；
- ApplicationMgr:Awake()被Unity引擎自动调用；
 - 调用ApplicationMgr:Initialize()，在这个函数中顺序调用了以下成员函数来进行初始化：
 - InitializeMode()：设置模式为ApplicationMode.PUBLIC；
 - InitializeUnity()：设置了UnityEngine.Application的一些属性；
 - InitializeGame()：看来核心的内容在这里，初始化Network，GameMgr等；
 - InitializeWindowTitle()；
 - InitializeOptionValues()；

游戏启动应该不只这点东西。由于我们纯靠程序集的动态分析，无法知道它的场景编辑、对象的脚本绑定，也不能跟踪调试，所以只能靠猜测了。另外，一些事件是通过网络消息触发的，这也给静态分析带来了一些难度。OK，我们继续。我注意到了class Login，它从Scene派生。查看了一下Scene的派生类还有不少，我猜测每个派生类，作为特定Scene逻辑处理的脚本。而Login应该是在第0个场景中被激活运行。我们看一下Login:Start()，这属于MonoBehavior自动调用，他主要做了这样几件事：

- 注册了一些资源版本检测、Login相关的网络消息回调；



- 通知SceneMgr场景加载完成；
- 调用成员函数：AutoLogin()；此函数调用Network.AutoLogin()；
 - 从配置文件中找到User Name，然后调用ConnectAPI.AutoLogin()——奇怪的是发现这个函数只是简单的返回false，并没有进行实际的操作。

我们在来看一下Login:Update()，这个也是属于被自动调用的脚本函数。在这里它检测了Login的状态，并调用了成员函数LoginOk()，而它有主要调用了AssetsVersionCheckCompleted()，这个函数内容很丰富：

- 通知其他模块，已经登录成功，包括：BaseUI、InactivePlayerKicker、HealthyGamingMgr、GameMgr；
- 调用一些模块的Initialize函数，包括：DefLoader、CollectionManager、AdventureProgressMgr、Tournament、StoreManager等；

我们前面看到了Login从Scene派生，并且还有一个SceneMgr类。我们可以断定游戏根据不同的逻辑划分成了一些scene，接下来我们就探索一下Scene切换的流程。还是从Login入手。

以下流程都是在Login类中完成，下面描述的过程都是Login成员函数的调用：

- 首先我找到了Login:OnNetCacheReadyStep2()函数，这个应该是login流程中某一步的网络消息回调函数；
- 它会调用 WaitForAchievesThenInit()这个Coroutine函数，这个函数检测了是否需要播放视频，然后调用ReconnectOrChangeMode()；
- 此函数处理重新连接，一般的话应该是调用了ChangMode()；
- ChangMode()处理了新手教程相关的启动逻辑，一般的话会调用ChangeMode_Hub()；
- 这个地方貌似是调用了技能特效，特效播放完成之后调用回调函数：OnStartupHubSpellFinished()；
- 此函数调用ShowUnAkedRewardsAndQuests()；
- 这里面主要是调用了HandleUnAkedRewardsAndCompletedQuests()，哦~，**这应该是游戏启动的时候显示当前任务还有未领取的奖励的那个界面；**
- 它会调用ShowNextUnAkedRewardOrCompletedQuest()；
- 其中主要调用了ShowWelcomeQuests()；
- 当幸亏显示的任务为0时，则调用了这一句：
SceneMgr.Get().SetNextMode(SceneMgr.Mode.HUB)，这是关键的一步了；

接下来我们就跳转到SceneMgr类中，继续探索Scene切换流程的实现。以下都是指的SceneMgr的成员函数：

- SetNextMode()函数主要就是把“m_nextMode”成员变量设置为了指定值；
- 接下来看一下Update()，这个函数主要是检测了是否需要切换Mode，然后调用了：
 - SwitchMode()：这个是一个Coroutine，它主要是调用了LoadModeFromModeSwitch()，它的核心也是调用LoadMode()；
 - 直接调用LoadMode()
- LoadMode()函数，主要是根据当前的Mode，调用LoadScene()；
- LoadScene()函数主要是调用了：**Application.LoadLevelAdditiveAsync(this.sceneName);**
- 这样就完成了场景的切换。

OK。通过以上分析，我们大体了解了游戏启动过程是这样的：

- 进行账户验证；
- 账户验证完毕之后，显示未领取的奖励和任务；
- 然后切换到SceneMgr.Mode.HUB模式，即加载了相应的Scene。

关于游戏启动的分析告一段落，下一篇将会分析一下炉石的Scene组织。

版权声明：本文为博主原创文章，未经博主允许不得转载。

▼ 下一篇 [《炉石传说》架构设计赏析\(2\)：Scene管理](#)

发布

- 深入浅出Unity3D——第一篇
- 基于Unity的游戏开发（下）
- Cocos2d-Lua手游开发基础篇
- .NET平台和C#编程从入门到精通
- 韦东山嵌入式Linux第一期视频
- 《炉石传说》架构设计赏析2Scene管理
- 《炉石传说》架构设计赏析7使用ProtocolBuffers
- 《炉石传说》架构设计赏析7使用
- 《炉石传说》架构设计赏析5卡牌&技能的静态数据
- 游戏架构设计与策划的基本流程是什么

更多职位尽在 **CSDN JOB**

10-17K/月

1-2K/月

[查看评论](#)

123

好

谢谢LZ

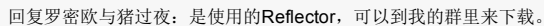
LZ, 程序集是指安装目录下的那些吗, 能具体说是哪个dll吗

回复kkxsi: Assembly-CSharp.dll、Assembly-CSharp-firstpass.dll, 主要是这两个


4/5

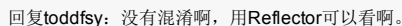


Re: 房燕良 2014-10-09 09:48发表

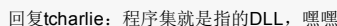


C

Re: 房燕良 2014-10-09 09:48发表 



C


Re: 房燕良 2014-09-15 16:40发表 

Re: 房燕良 2014-09-10 21:06发表



* 以上用户言论只代表其个人观点, 不代表CSDN网站的观点或立场

全部主题	Hadoop	AWS	移动游戏	Java	Android	iOS	Swift	智能硬件	Docker			
OpenStack	VPN	Spark	ERP	IE10	Eclipse	CRM	JavaScript	数据库	Ubuntu	NFC		
WAP	jQuery	BI	HTML5	Spring	Apache	.NET	API	HTML	SDK	IIS	Fedora	XML
LBS	Unity	Splashtop	UML	components	Windows Mobile		Rails		QEMU	KDE	Cassandra	
CloudStack	FTC	coremail	OPhone	CouchBase	云计算	iOS6		Rackspace	Web App			
SpringSide	Maemo	Compuware	大数据	aptech	Perl	Tornado		Ruby	Hibernate	ThinkPHP		
HBase	Pure	Solr	Angular	Cloud Foundry	Redis	Scala	Django		Bootstrap			

 网站客服
  杂志客服
  微博客服
  webmaster@csdn.net
  400-600-2320 | 北京创新乐知信息技术有限公司 版权所有 | 江苏乐知网络技术有限公司 提供商务支持

京 ICP 证 070598 号 | Copyright © 1999-2014, CSDN.NET, All Rights Reserved