

## 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: 125162次

积分: 1743

等级: BLOG &gt; 4

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



## 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(2): Scene管理

分类: Unity3D

2014-09-13 11:58 7041人阅读 评论(6) 收藏 举报

unity3d

游戏

欢迎来到我的酒馆, 快来火炉旁暖暖你的靴子。哈哈, 我们继续欣赏炉石的代码。欢迎转载, 请注明作者【燕良@游戏开发】及原文地址: <http://blog.csdn.net/neil3d/article/details/39231541>

上篇文章我们分析到Scene的加载工作, 今天我们要分析一下炉石这款游戏中一共有哪些Scene, 他们各自负责什么逻辑、UI的处理方式。

在正式开始之前, 我来对Scene切换再做一些补充分析。前文中我们看到SceneMgr是调用了“Application.LoadLevel(sceneName);”, 那内存中的东西岂不是越搞越多吗? 我们再仔细看一下SceneMgr:SwitchMode函数, 它是一个Coroutine, 他主要进行了下面这几个步骤的操作:

1. 调用当前Scene的Scene:PreUnload()函数;  
发送FireScenePreUnloadEvent事件;  
等待直到Unload过程走完(通过检测LoadingScreen的阶段);
2. 调用Scene:Unload()函数;  
发送FireScenePreUnloadEvent事件;  
调用成员函数PostUnloadCleanup()函数, 它调用了两个关键的函数:
  - 首先是成员函数: DestroyAllObjectsOnModeSwitch(), 这个函数查找到所有的GameObject (Object.FindObjectsOfType(typeof(GameObject))), 然后进行了筛选(通过成员函数ShouldDestroyOnModeSwitch), 除了一些全局对象之外(主要是SceneMgr、PegUI、Box、DefLoader), 全都删除了(通过调用Object.DestroyImmediate());
  - 然后调用了: Resources.UnloadUnusedAssets();
3. 然后是调用前文提到过的成员函数: LoadModeFromModeSwitch(), 进行了LoadLevelAdditiveAsync操作;

综上所述, 炉石的Scene切换主要是包含两步: 1删除所有非全局对象, 卸载未引用的Asset; 2加载新的Scene。(我倒是想到另外一个土鳖一点的替代方案: 创建一个完全空的scene, 调用LoadLevel加载它, 那么所有没有设置"DontDestroyOnLoad"的对象就都被删除了。)

除了前文提到的Login, 我们可以看到Scene还有很多派生类, 详见下图:

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转

(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: 解压缩看到一个

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/detai...

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。

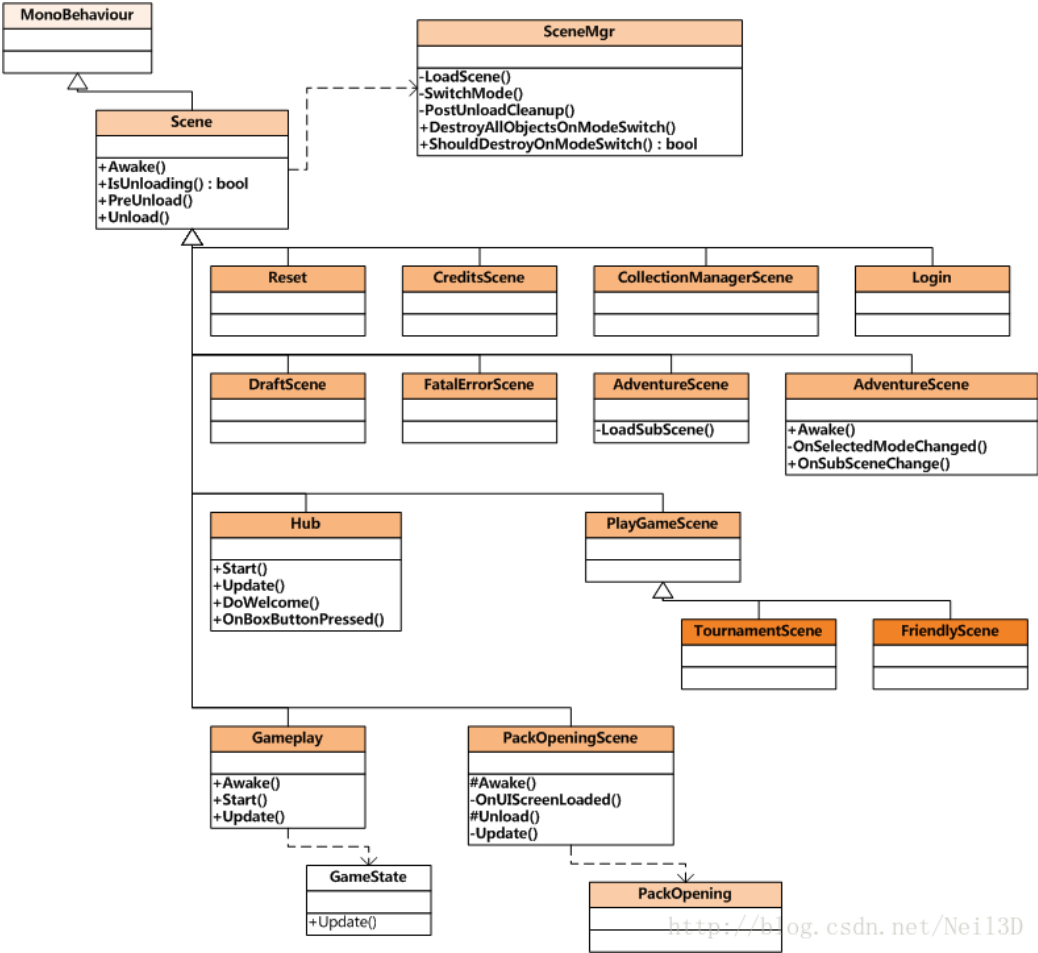
(๑\*W\*๑)

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



这是我猜测的这些类和游戏内容的对应关系，没有太仔细分析，可能有些对应是错误的：



下面我们就挑选一个简单的Scene来分析一下它的内部运作机制，我们来看一下AdventureScene吧。

Adventure相关的Class很多，我们只做一个粗略的分析，只涉及到下面这几个类和接口：



首先我推测，在Hub屏幕中点击中间的【Solo Adventure】（冒险模式）按钮之后，通过我们前文分析的LoadScene流程，加载了一个冒险模式相关的Scene。它里面有一个GameObject绑定

了“AdventureScene”这个脚本，我们可以看到AdventureScene:Awake()方法中主要是注册了很多事件的回调。

我们可以看到有一个“AdventrueSubScenes” 枚举，它基本上对应了下图中的按钮：

```
[csharp]
01. public enum AdventureSubScenes
02. {
03.     Chooser,
04.     Practice,
05.     MissionDeckPicker,
06.     NormalHeroic,
07.     ClassChallenge
08. }
```



接下来还有一个"AdventureSubScene"是处理子场景对应的一些逻辑的。  
我们看到有 "AdventureChooserTray" 这个类：

- 我推测这个类就是用来处理上面这个游戏画面的UI交互操作的；
- 这个类在Awake时，通过调用 "CreateAdventureChooserButton()" 方法创建了上图中的上部分那几个冒险游戏内容模式相关的按钮；
- 这些按钮都绑定了事件回调：AdventureChooserTray.ButtonModeSelected(); 当这些按钮被点击时，主要是调用：
  - AdventureConfig:SetSelectedAdventureMode(), 此函数修改内部数据之后触发事件：FireSelectedModeChangeEvent()
  - AdventureChooserTray通过OnSelectedModeChange()响应此事件，这也就是点击上面那几个按钮之后要做的一些处理：包括更新左侧的画面、设置Choose按钮状态等等；其中调用了PlayMakerFSM，主要是向其发送事件 "Burst" ；通过这里，我们可以确定炉石使用了PlayerMaker插件。
  - AdventureScene也通过OnSelectedModeChanged()相应了此事件；
- 它里面还有一个 "PlayButton m\_ChoseButton" 成员变量，并把它添加了EventListener，用来调用ChangeSubScene()方法。这就和游戏实际的操作对应上来：在上面选择模式，然后点击下面的【Choose】按钮，就进行到下一步的选择了。
- AdventureChooserTray:ChangeSubScene()通过Coroutine的方式调用了AdventureConfig:ChangeSubSceneToSelectedAdventure(), 然后调用了AdventureConfig:ChangeSubScene(); 它主要触发两个事件：
  - FireSubSceneChangeEvent：AdventureScene通过OnSubSceneChange()函数响应它，主要是调用AdventureScene:LoadSubScene(), 内部主要是调用AssetLoader.LoadUIScreen();
  - FireAdventureModeChangeEvent：AdventureScene通过OnAdventureModeChanged()响应它。

通过上面的分析，我们大致了解了上面这个游戏截图中的操作实现逻辑。  
这次的分析算是一次热身，接下来重点分析有两个方面：

- 游戏逻辑的组织，特别是技能的数据、逻辑组织；这可能需要经过多次尝试，慢慢接近；
- 游戏的Asset资源管理、加载机制；

OK，今天的分析就到这里，欢迎大家拍砖。后续分析敬请期待！  
顺便来秀一下我的鱼人部队，别看这些1点学的小东西，加在一起还蛮欢乐的！





版权声明：本文为博主原创文章，未经博主允许不得转载。



- ▲ 上一篇
- ▼ 下一篇
- 《炉石传说》架构设计赏析(1)：游戏启动流程
- 《炉石传说》架构设计赏析(3)：Gameplay初探

顶

7

踩

1

主题推荐

- ui
- 架构设计
- 游戏开发
- 管理
- 工作
- 代码

猜你在找

- iOS即时通讯(IM)开发实战篇-基于XMPP的聊天软件开发
- 《炉石传说》架构设计赏析7使用
- 基于Unity的游戏开发（下）
- 《炉石传说》架构设计赏析5卡牌&技能的静态数据
- Cocos2d-Lua手游开发基础篇
- 《炉石传说》架构设计赏析7使用ProtocolBuffers
- 深入浅出Unity3D——第一篇
- 《炉石传说》架构设计赏析6卡牌&技能数据的运行
- Python编程基础视频教程(第三季)
- workflows管理系统-软件架构设计

准备好了么？跳吧！

更多职位尽在 CSDN JOB

■ 020平台架构师	我要跳槽	■ J2EE架构师	我要跳槽
广州普及网络科技有限公司北京分公司	20-30K/月	福州中森网络科技有限公司	6-12K/月
■ cocos2dx高级开发工程师	我要跳槽	■ 产品经理（后台管理）	我要跳槽
北京火谷网络科技有限公司	10-20K/月	钰诚国际控股集团	20-35K/月



XamarinWrite C#. Run on 2.6 billion devices.Download Free Trial

查看评论

4楼ZeroDeep2015-08-12 16:28发表

QQ群多少啊

3楼青蛙果2015-02-05 11:10发表

我是c++初级吧，问个可能比较傻的问题吧。  
在这个游戏中，自己可以保存的套牌最多有九套，这个从一开始就被很多玩家吐槽了，希望增加卡槽。据说开发组表示增加卡槽非常的困难。这个我不太理解，结构不是完全相同吗？为何会很困难。自己的套牌也不涉及和其他玩家的交互问题，这个我很困惑。单纯的增加记录也不应该会占用非常大的存储空间吧，最多就是（30 + 1 + 1）\* sizeof(DWORD)吧。

2楼zephyr11252014-10-06 14:51发表

AdventureSubScenes的枚举应该不是对应那些按钮的。而是对应冒险模式下的几种子界面：  
Chooser - 冒险模式选择，  
Practice - 练习模式难度选择，  
MissionDeckPicker - 冒险用的牌组选择，  
NormalHeroic - 普通/英雄模式关卡选择，  
ClassChallenge - 职业挑战模式关卡选择。

Re: 房燕良2014-10-07 11:31发表

回复zephyr1125：有一起研究的了，太好了。有空到我的QQ群里来交流一下吧。

1楼Golden\_Shadow2014-09-29 13:44发表

哈哈.... 分析得好.另外,还真是会玩啊...

Re: 房燕良2014-09-29 17:12发表

回复Golden\_Shadow：做游戏的嘛，多少也要会玩一点。：)

您还没有登录,请[\[登录\]](#)或[\[注册\]](#)

\* 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

核心技术类目

全部主题

HadoopAWS移动游戏JavaAndroidiOSSwift智能硬件Docker

OpenStackVPNSparkERPIE10EclipseCRMJavaScript数据库UbuntuNFC

WAPjQueryBIHTML5SpringApache.NETAPIHTMLSDKIISFedoraXML

LBSUnitySplashtopUMLcomponentsWindows MobileRailsQEMUKDECassandra

CloudStackFTCcoremailOPhoneCouchBase云计算iOS6RackspaceWeb App

SpringSideMaemoCompuware大数据aptechPerlTornadoRubyHibernateThinkPHP

HBasePureSolrAngularCloud FoundryRedisScalaDjangoBootstrap

公司简介 | 招贤纳士 | 广告服务 | 银行汇款帐号 | 联系方式 | 版权声明 | 法律顾问 | 问题报告 | 合作伙伴 | 论坛反馈

 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 



http://blog.csdn.net/neil3d/article/details/39231541

5/5