[Cocos2d-X 论文转载](http://blog.csdn.net/oskytonight/article/details/8263112)

2012-12-06 01:00 1124人阅读 评论(0) [收藏](javascript:void(0);) 举报

[论文](http://www.csdn.net/tag/%e8%ae%ba%e6%96%87)

　　3月 31 日消息，人人网游戏工程师李成在第四届 Cocoachina 开发者大会发表题为“基于 Cocos2d-X的游戏框架设计”的主题演讲。

**以下为演讲实录：**

　　大家好！

　　我是来自人人游戏的李成，我今天演讲的题目是“基于 Cocos2d-X的游戏框架设计”，大家有什么好的想法可以跟人人游戏相关的同事交流。

　　我为什么要讲这篇 PPT，我同学在 Cocos2d-X开发的时候会发现网络上面有大量的例子，但是没有把技术点连起来作为一个整体经验的设计。今天大家很多都是讲 Cocos2d-X平台的移植性，我今天讲的是怎么针对 Cocos2d-X平台做游戏类的开发。在前期希望大家有一些更好的设计思想，避免在项目的中后期遇到难以扩展或者是难以复用的问题。我个人经历过 PC 端类游戏的开发，和 Web 类游戏的开发，所以我感觉端内游戏和页游传统游戏的开发经验完全值得借鉴，运用到移动游戏的开发过程中。

**本次演讲主要分为三部分：**

　　第一，软件设计不仅仅是游戏开发，它跟传统的建筑业有很直接的关系。你可以从两方面互相汲取一些经验。在盖大楼的时候，如果楼踏了就相当于游戏崩溃了，这种两种情况会直接导致什么结果呢？如果楼踏了很多人就必须重新盖，相对于程序员来说，就必须重新返工，很多其他项目的人员都一直跟着你加班加点。如果你把游戏做崩溃了，项目没有成，并且没有女朋友的话，别人都不愿意跟你，如果有女朋友的话你丈母娘就不愿意，说你把游戏都整崩溃了，还有什么不敢干的。

　　简单总结一下，游戏开发跟盖大楼有一些共同点。一是前期设计规划很重要；二是基础设施、基础模块的构建很重要；三是扩展性、重用性方面很重要；四是健壮性、安全性很重要；五是从基础做起从细节做起，拒绝豆腐渣工程。

　　第二，主要讲一下 Cocos2d-X跟 ios 平台是怎么样结合在一起的。

　　先介绍一下 Cocos2d-X引擎开发的优势，因为 Cocos2d 的 C++ 版本，目前有大量 Cocos2d 的经验分享，所以如果是 Cocos2d 的 C++ 版本的话，可以把之前的经验分享过来。C++版本对于我这种非出身的人来说可能是一种福音。支持 ios、Android、Windows 等平台，跨平台开发者的福音。它是开源免费、易学易用、庞大的工具链支持；早上还有刚才很多同学分享了一些工具。还有强大的技术支持，活跃的技术社区交流平台，包括我也有 QQ 群，那个 QQ 群经常闪，导致我工作的时候不得以把 QQ 都得关掉。多款的线上游戏应用经验，如果一个东西没有被证实过，我们盲目使用做一个项目的话，可能风险是非常大的。Cocos2d-X不断完善和改进，逐渐增加更多的新技术，比如说跟 HTML5 方面的结合。

　　介绍一下 Cocos2d-X引擎框架图，早上我以为作者会讲这块儿，我把这块儿补上。Cocos2d-X整体框架图整个引擎有一个导演，在引擎模块导演有场景的概念，场景上可以挂很多的层，游戏表现都在层上做，在层上可以再加一些特效，再加一些效果，这样就构成了动态的画面，再在动态的画面上做自己的游戏开发逻辑组成一个产品。

　　下面简单介绍一下 ios 应用程序的框架，因为这篇 PPT 主要是讲 Cocos2d-X跟 ios 框架怎么结合的，所以对于 ios 框架本身是什么的机制非常重要。首先，ios 的应用框架最简单的主要类就是六个模块，五有一个整体客户端，这里面主要是做程序的初始化，还有消息的响应、循环，接下来是 Delegate 是对外围的扩展，在关键点通过回调的方式，让我们知道现在游戏开始加载，现在游戏进入后台，这个游戏要关闭了，通过这个可以动态的及时的获取游戏的运行状态，做出一些调整。接下来是 UIScreen，就是屏幕大小可以通过这个来获取；下面是 UIWindow、UIView，这个窗口很直观，一个游戏至少有一个 UIWindow，会有一个消息响应，会把消息响应放在 UIView 上面，View 也有一个控制器，控制 View 的运行状态。

　　ios 这个框架图，介绍一下消息的响应，通过消息发送给 UIWindow，再上升到各个 UIView，这是整体框架。整体的运行周期，刚开始有一个初始化，消息注册这类的事情。如果你被打断会失去焦点，要做什么呢？就要把音效去掉，否则接电话的时候音效还在播放就不人性化。退出的时候要要是你游戏要退出了，要做资源的保存，游戏的清除。还有几个存在的状态，在前台还是后台，在这些状态中要对游戏进行相应的处理，否则的话接电话的时候就非常怪，接电话的时候游戏音效还在播放。

　　接下来介绍 Cocos2d-X跟 ios 是如何结合的？ios 主要是 View 和 UIWindow，所以一个平台必须有 View，如果没有 View 否则就没有办法表现任何东西。客户端最主要的 View 界面是捕捉、风和分发系统 TOUCH 事件。外层的 RootViewTontroller 为其控制器，可通过该客观器对 EAGLView 进行相关的控制。OpenGL-ES 无进行渲染更新。

　　介绍引擎的总体更新流程，对游戏开发者来说我们必须游戏是怎么运行的，每个运行状态是什么。一般来说，我们进行了 UIView 初始化完全后，会通过 Delegate，初始化 EAGL View 会调用引起 CCapplication 的 run（）接口。一般的逻辑更新会放到 CCDirector 内部，调入之后才会进行熏染更新。第一步进行逻辑更新，第二步进行客户端的渲染更新。

　　第三、游戏客户端总体架构篇

　　简单介绍一下客户端游戏开发的总体架构，这张图以分层分模块的方式介绍游戏开发架构中分哪些东西？我们一般会 Cocos2d-X引擎库、会有音效的播放库，如果有自己的基础公共库也可以放到上一层。要用到简单的解析工具，接下来是脚本。客户端主要的模块划分，一般客户端都需要这么多模块，比如说输入输出的模块。游戏客户端一般会面临大量的数据，包括资源数据、道具数据、音效数据、关卡数据，所以我们提出了数据层，主要是对静态资源进行统一化的管理，比如说加载、卸载、获取接口之类的事情。另外，就是日志的模块，开发过程中，一般要记的日志非常多，如果日志通过一种方式记文件的话，可能效率也不太好。如果上线的话，大量的数据是不需要的，所以我们在日志模块进行了分类，分开关的方式，可以分几档日志，一般调试的日志放到一个文件里面去，错误的日志放到一个地方，引发崩溃的日志放到一个地方，这样在开发的时候是非常方便的，日志是分开的，而不是一团糟，也不方便我们记录一些问题。在上线的时候可以把这些日志开关关掉以提升性能。

　　网络层一般对网络，这里使用的原生的开发，必须对原生的进行封装，下面我们会有详细的讲解。在这么多的基础模块之上，再做游戏逻辑，再到下一个层，除了游戏逻辑层其他的模块都可以复用，这是复用性很强的总体框架设计。

　　客户端主模块的设计：首先我们使用单一的 CCScene，因为它可以支持多个 Scene 的跳转，我们拒绝不必要的花哨表现，以简化代码实现。基于帧率的游戏更新处理，比如说客户端的 MINI 无 Finger 集成自 CCNode，且在初始化 Init 函数中通过 schedule 设置游戏逻辑主更新函数 Tick（），以此确保每次渲染之前，都会首先进行逻辑更新处理。完全基于帧率的定时器调度，所有的定时器处理，完全依赖于游戏帧率，简单直接。网络模块总体设计图，网络层分为 UDP 的方式，在网络程序设计中是很著名的设计模式。大致的意思是将不同块儿的内存节点，以池的方式放在一个队列中，这种方式是非常快速跟简捷的。

　　原生的 SOCKET 封装，原生的 socket 编程，简单直接，代码可控方便定位问题。Non-Blocking I 无O使用费组塞式 IO 模式，无需开辟网络线度对象 TCP 来说，简单的连接线程。Selector 轮训模式，监听所有处理。

　　TCP&UDP 具体的区别不在此多说了，主要说一下对于 TCP 来说，为了解决年糕问题，因为 TCP 是自节流的方式，所以存在年糕问题，对于年糕问题技术代码控制有一个方式，MemNode 来接收数据。对于 UDP 来说，因为它本身是无序的，会使用 Sequence Number 方式处理杂乱，无序包。

　　定制内存池管理器：绝大部分的数据包大小限制于 4096bytes；Free-List 内存池技术，NO-MEMCPY；避免频繁 new/delete。

　　异或加密方式：网络游戏在服务器有一个非常强的校验规则，如果有异常服务器就会强制断开，所以我们要做一些简单的异或加密。一次异或明文变密文；二次异或就是密文变明文。

　　仅指游戏数据层：非 ios 常规开发中的数据层，特制游戏内的静态数据层。

　　客户端通常需要大量的静态资源：图片资源&音效资源，窗口布局文件；其他游戏策划数据。

　　拒绝扁平化文件存储，一个文件放一个文件的方式，我们要拒绝这种方式。推荐使用 SQLite，在移动平台上推荐使用轻量级的数据库。对静态数据进行统一读写，提高 IO 读取效率。

　　统一管理：建立统一的加载，更新和卸载登记制；方便游戏逻辑管理和监控。因为我们可以监控每一个模块，是不是内存超了，服务器保存的内存是不是过大，时时刻刻对内存有一个监控。

　　数据模块示意图：上面每一个具体的数据管理器都会从 DB 开发，这上面生成了很多音效、关卡、道具等各个数据的管理器。

　　UI 布局示意图：在客户端里面最底层做一个 Layer，做重力感应的事件，对这些事件进行统一管理派发。在根窗口上再做游戏界面布局，比如说现在有一个游戏布局文件，这也是一个 Layer，在这之上有多个子 Layer，有控制区域和面板，组成树型结构，构成了整个游戏的 UI 布局。

　　下面简单介绍一下 UI 系统模块：

　　窗口布局组：相同功能 UI 组建合并为一个布局 Layout，统一化管理：初始化/加载/更新/事件响应/卸载操作；基于消息事件的处理方式，方便脚本拓展；采用外部配置处理/支持动态化配置。

　　独立的根窗口（UILayer）：系统最底层的窗口层，所有 Layout 的父窗口，游戏内唯一的监听 Touch，Accelerometer 事件的 CCLayer，采用用户输入，统一化处理。

　　脚本拓展：Lua 语言拓展，将具体逻辑和游戏框架分离，加强健壮性。

　　UI 布局配置化文件示意图：现在开始 UI 的布局，下面有一个 Window，创建一个 Layer，在游戏中会创建一个 CCLayer，名字叫什么，颜色、位置、大小之类的事情。在这个之上要做一个 Tab，在上面也可以初始化。如果做知识脚本处理的话，这个地方可以写清楚对什么事件执行什么处理。

　　游戏 UI 系统模块：关于模态窗口，修改源代码，在 CCNode 中增加优先级的概念，接着增加父窗口的优先级。所有添加的 UI 组建，需要明确设定是否继承父节点的优先级，以此形成一个优先级响应队列，从而实现模态窗口的功能，借助系统 UIView，实现单独的 View 界面，附加在 EAGLView 之上。

　　相关优化：所有 Layout 延迟加载，不使用时马上卸载，释放内存，合并需要的图片资源。

　　游戏音效模块：这个是大家比较热心的模块，我推荐使用 FmodEx，成熟高效，接口简单统一，无需考试平台化差异；功能强大，支持 3D 音效，静音、暂停、音量大小等设置完全满足日常的音乐开发；当然需要购买。引用计数：相同的音效仅需要公用一份资源数据；及时清除不需要的音效资源，减少内存占用。

　　人性化设置：增加人性化的设置面板，提升用户感受；提供动态开启/关闭，设置音量等功能；按照用途，划分音效类型，单独管理；通过 Application Delegate，程序获得焦点时播放音效，失去时静音。关于音效资源，音效文件尽可能的小，降低内存占用。

　　消息事件管理模块：整个客户端基于消息事件驱动；实现和调度分离，最大程序降低耦合。

　　调试器管理器模块：XCODE 自带的调试器工具不够用；需要针对渲染、网络、逻辑数据进行更细化的监控；两者结合，宏观微观，两手一起抓。

　　日志模块：日志分等级，分输入方式，完全可配置；不同类型日志，按照等级分开记录。

　　还需要什么？消息推送：借用本地和远程的消息推送机制，提高产品年合度；借助“社会工程”调动玩家积极性；什么叫“社会工程”呢？做杀毒软件和做木马的人对这种东西非常理解，比如说我昨天我邮箱还收到社会工程的引导，说什么孤独少女寂寞加好友有一个链接，用这种挑逗性的东西引导你，你一点你的电脑就挂了。移动产品的 UI 界面设计，移动产品的 UI 界面尽量简单直接，个性化操作。移动产品的特性：考虑移动产品的特性和人们的使用习惯等，有针对性的设计。

我的演讲到此结束。谢谢大家。

霍常伟;[基于cocos2d-x引擎的移动游戏UI系统设计及应用](http://cdmd.cnki.com.cn/Article/CDMD-10004-1012355744.htm)[D];北京交通大学;2012年

舒沂;[基于cocos2d-x引擎的手机游戏快速开发工具的设计与实现](http://cdmd.cnki.com.cn/Article/CDMD-10284-1013190510.htm)[D];南京大学;2013年

爆破炸药能炸掉石头钻石黄金但炸不掉碎金；

力量源泉

幸运草使宝袋出现宝贝几率增加，但试验发现钱出现少了炸药出现最多再次力量

炒石妙语3倍增值

钻石升值器增值1.5倍