<Catcha Mouse>

<Catcha Mouse Document>

Version <1.4.3>

文档信息

|  |  |
| --- | --- |
| 标题： | Cactha Mouse |
| 文件位置： | C:\Users\User\Desktop\Catcha Mouse Document |
| 版本： | 1.4.3 |
| 提交人： | <SJTUZWJ> |
| 提交日期： | <27/7/2018> |
| 状态： | 封闭测试 |

修订历史记录

|  |  |  |
| --- | --- | --- |
| **日期** | **版本** | **说明** |
| <4/7/2018> | <1.1.0> | <PVP基本玩法> |
| <5/7/2018> | <1.1.1> | <音乐音效> |
| <6/7/2018> | <1.1.2> | <代码重构+音量调节> |
| <7/7/2018> | <1.2.0> | <PVE+等级机制+皮肤+难度设置> |
| <8/7/2018> | <1.2.1> | <EVE+超时惩罚+角色选择+机器角色扮演> |
| <9/7/2018> | <1.2.2> | <接口重构+存读档本地化+记录复盘> |
| <11/7/2018> | <1.2.3> | <记录数据本地化+设置数据本地化+撤销功能> |
| <12/7/2018> | <1.2.4> | <分离数据结构与操作+重构Prototype,统一接口 > |
| <13/7/2018> | <1.3.0> | <隐身信息不对称玩法,装饰器模式重构继承树 > |
| <15/7/2018> | <1.3.1> | <修复存读档导致的bug> |
| <16/7/2018> | <1.3.2> | <备忘录模式重构记录> |
| <17/7/2018> | <1.3.3> | <伪外观模式重构客户端> |
| <18/7/2018> | <1.3.4> | <工具函数类,重构场景接口,  老鼠动画效果> |
| <19/7/2018> | <1.3.5> | <粒子效果实现天气+完善音效> |
| <20/7/2018> | <1.4.0> | <C++新特性提高代码可读性+添加小彩蛋> |
| <22/7/2018> | <1.4.1> | <拆分Judge类+观察者模式重构Graph的只写接口> |
| <24/7/2018> | <1.4.2> | <确保单例模式线程安全+模板方法模式重构角色接口> |
| <25/7/2018> | <1.4.3> | <修复了全新配置下进度条进度非法的bug+完成release模式的迁移+完善经验系数> |

目录

1. 简介 5

1.1 目的 5

1.2 范围 5

1.3 定义、首字母缩写词和缩略语 5

1.4 参考资料 5

2. 软件架构 5

3. 静态设计 6

4. 动态设计 10

<Catcha Mouse>

# 简介

# 目的

明确迭代的版本信息,展示开发流程,描述和规定了软件设计和实现细节，介绍软件的代码结构与运行时刻的结构。

遵循原则 *Agile Development, Rapid Iteration, Refactoring and Design Patterns*

提高软件开发过程的能见度;记录开发过程的相关信息;提高开发效率；作为开发人员在一定阶段的工作成果和结束标志;便于潜在用户了解软件的功能、性能等各项指标。

## 1.2 范围

游戏 Catcha Mouse

## 1.3定义、首字母缩写词和缩略语

UI：user interface 用户界面

## 1.4 参考资料

|  |  |
| --- | --- |
| 文件名和位置 | 标题 |
| <http://docs.cocos.com/cocos2d-x/manual/zh/> Cocos2d-x 用户手册 | Cocos2d-x 用户手册 |
| <https://blog.csdn.net/qq_27650777/article/details/72724461> UML常用图的几种关系的总结 | UML常用图的几种关系的总结 |
| 书架  设计模式·可复用面向对象软件的基础 | 设计模式·可复用面向对象软件的基础 |

# 2．软件架构

本软件由Controller，Model，View，Protocol四个模块组成，Controller模块是应用程序中处理用户交互的部分，Model模块是应用程序中用于处理应用程序数据逻辑的部分，View模块是应用程序中处理数据显示的部分，Protocol模块提供程序共同遵守应用的信息。

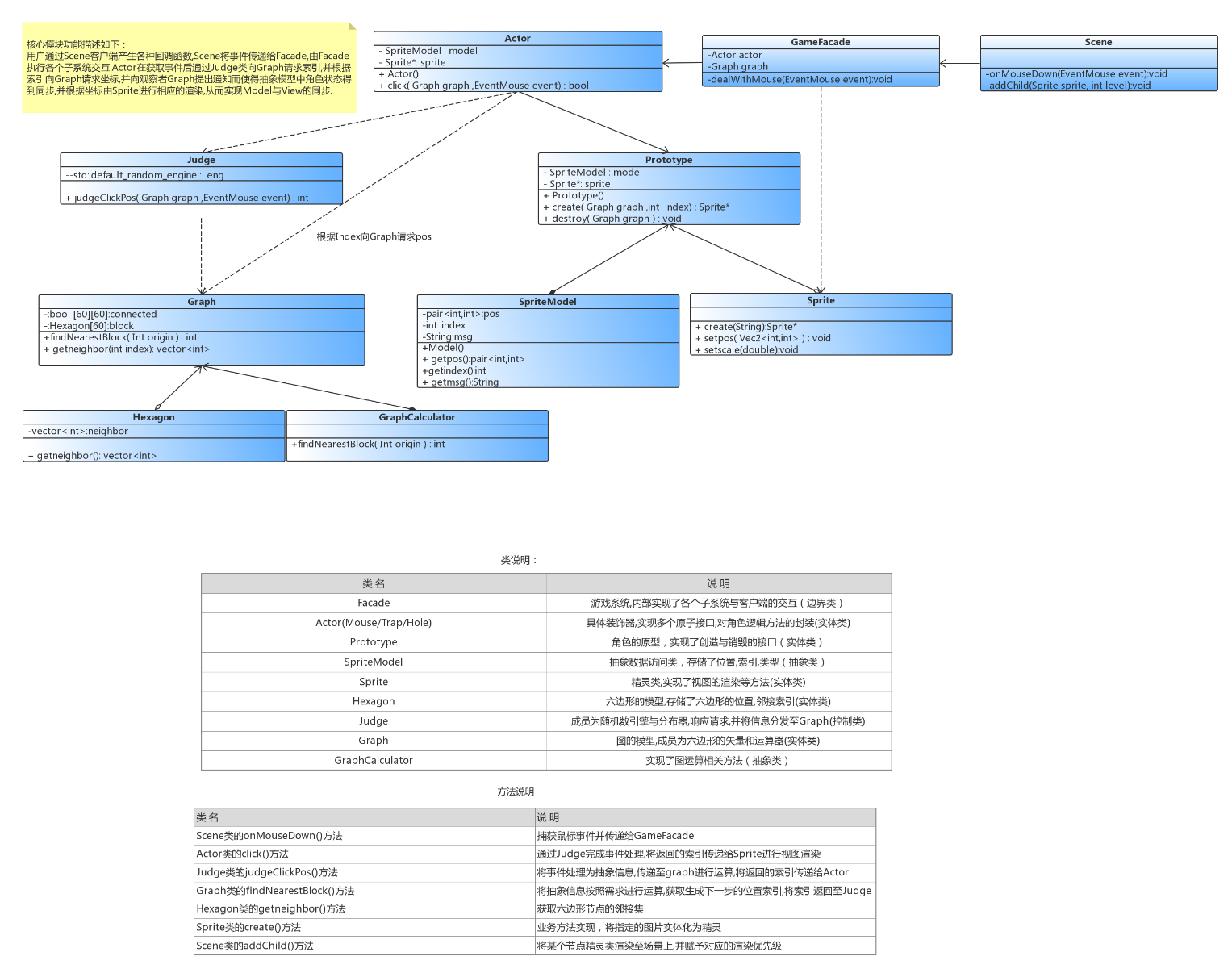
Controller模块。Judge进行图形逻辑的处理与随机事件处理；Referee进行胜负判断；DataController进行数据存储的处理; RecordController进行记录存储的处理;

View模块。Client负责进行场景的渲染； Actor子系统对游戏角色的数据和视图进行了封装;GameFaçade作为游戏系统的外观,处理各子系统间的交互;Asistant提供了视图所需功能组件。

Model模块。SpriteModel;负责存储游戏元素的抽象信息；Graph子系统对游戏的地图模型进行了封装。

Protocol模块。Bgm作为工具函数库，提供控制音乐音效的可复用函数； Setting负责存储并传递游戏全局共享的信息。

# 

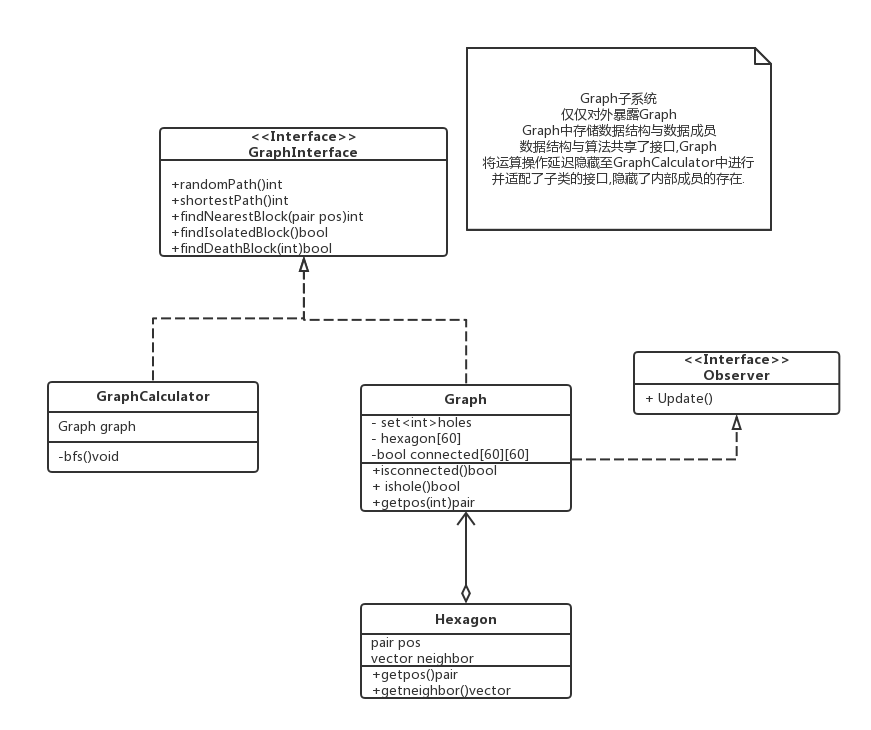
3.静态设计

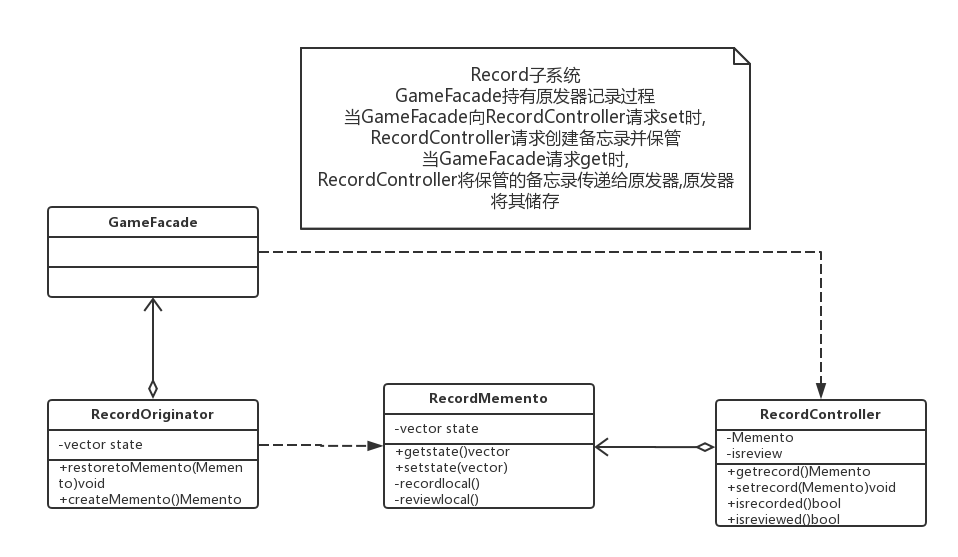
整体代码结构

(太大了,ctrl加滚轮放大看吧)

# 

Actor子系统



Graph子系统

Record子系统

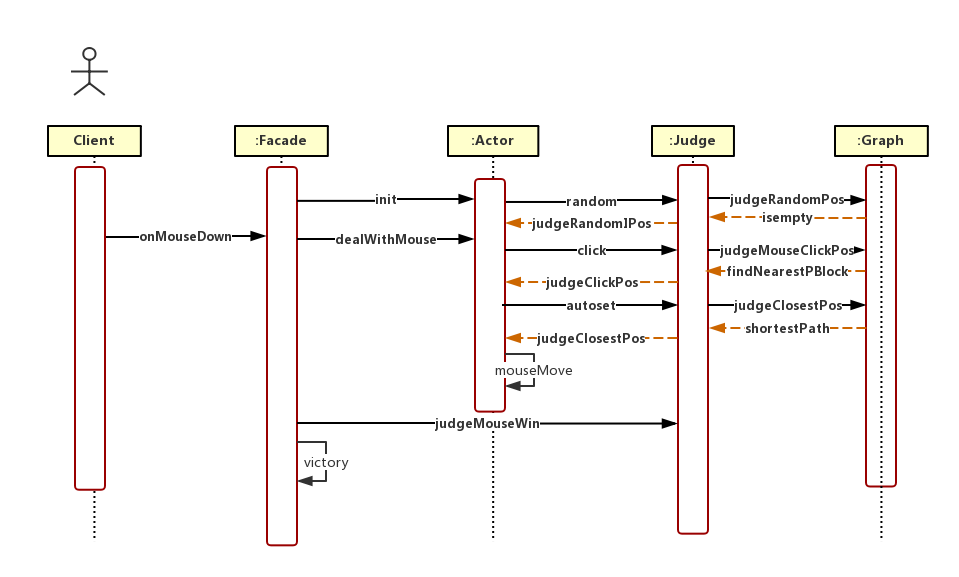
以上是游戏系统的三个子系统,具体描述见注释

其他子系统如Judge,Setting,BGM,DataController,Referee,

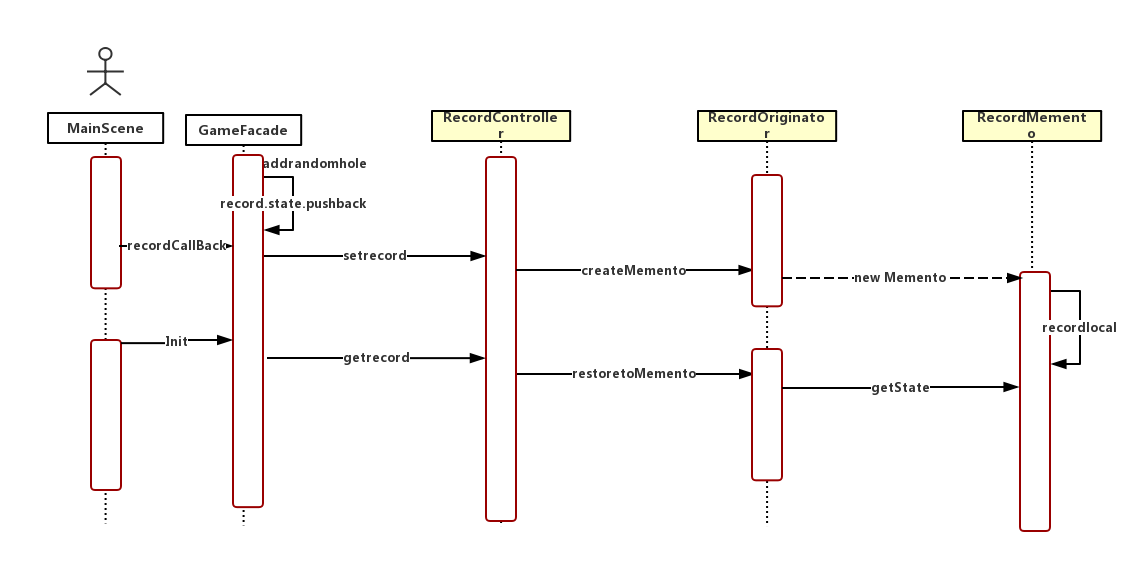
由于只有一个类于此省略

除子系统外,由Scene派生出的继承树作为客户端,定义了各种事件的回调函数,但是由于不涉及游戏本身逻辑,而只是对数据文件等增删查改的操作,因此于此省略.

4.动态设计

游戏机制 以PVE为例

由Client发出事件请求,由Façade进行交互处理,传递给Actor子系统,Actor将事件传递给Judge子系统进行初步处理,Judge将图形运算部分传递给Graph子系统,并将最终产物传递给Actor,由Actor负责渲染,并通过观察者接口使图同步.同时Façade也可以直接向Referee进行请求,判断胜负,根据状况而做出相应的行为.Façade的所有行为都是以Client作为舞台进行的



记录机制

由Client获取事件并传递给Façade进行交互处理,Façade向RecordController发出请求,最终产生备忘录并存储至本地.

如未退出游戏,备忘录的生命周期持续,可在下一次Façade请求获取备忘录时拷贝给原发器.

若退出游戏,Memento将会执行reviewLocal操作,并将产生的备忘录拷贝给原发器.

记录用于实现回放撤销等功能.