SET Game 设计说明

一、 综述

此为 SET Game 项目的设计说明文档，说明了 SET Game 项目开发中使用的设计框架、概念、代码结构与组成等。其主要目的在于对代码阅读和审查者进行引导，提高代码可读性。

特别指出，本文并非项目设计文档，仅为项目迭代完成时的说明文档。

项目完整代码位于 https://github.com/curno/SetCardGame/tree/master

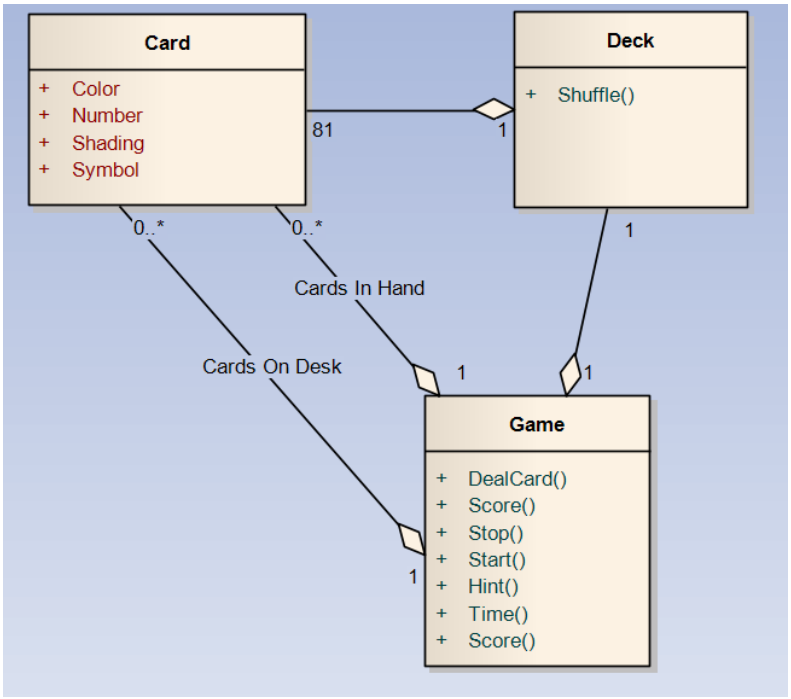
二、 总体设计

根据需求，本项目使用 MFC 作为窗口基本框架库进行开发。为了获得良好显示效果，使用 OpenGL 渲染游戏场景。使用 Apache 2.2 搭建私有服务器，使用 python2.7 开发 CGI 程序，并使用 MySQL5.5 作数据存储。

下面几小节将针对项目开发中的多个方面进行逐一的设计介绍。总体来讲，项目设计采用 MVC 架构，将模型（逻辑设计）和显示的耦合降到最低。

三、 逻辑设计

逻辑设计主要负责设计游戏中的核心部件。类图如下：

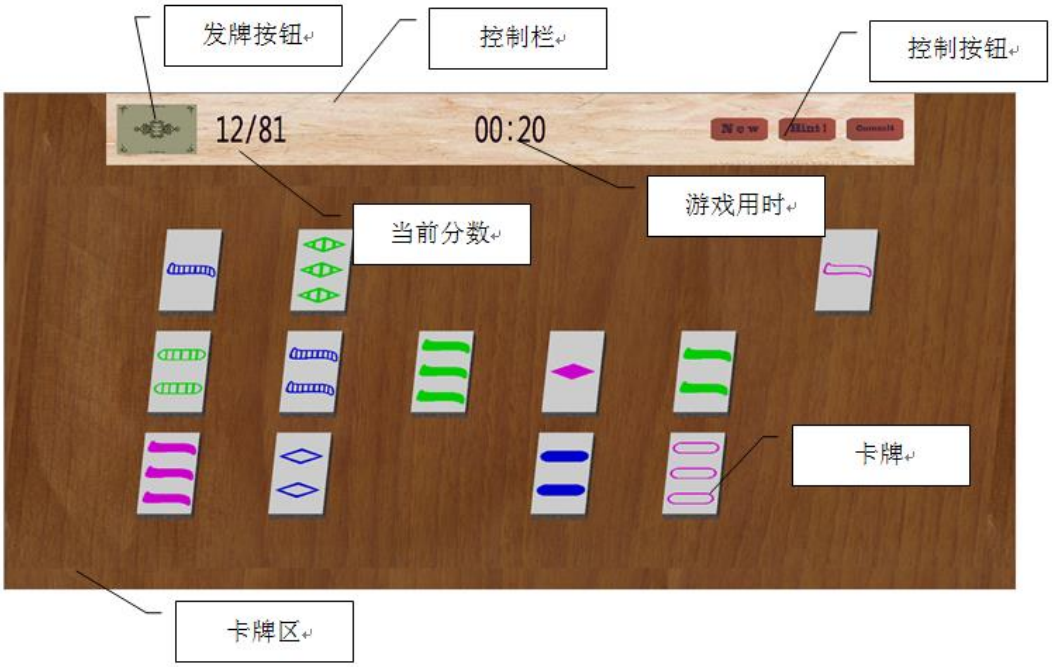


本部分由三个类组成，Card 为一张卡牌，有自己的四个属性；Deck 为一副牌，由 81 张卡牌组成，拥有洗牌方法；Game 为一次游戏，包含一副卡牌，同时，Game 还记录了当前游戏进度下桌面上的卡牌和尚未发出的卡牌，游戏计时和分数等，Game 还负责游戏开始、结束、提示等逻辑。

本部分代码位于工程的 Include/Model 下。

四、 界面设计

界面设计说明如下图所示：



关于卡牌的图案，使用手工绘制的图片贴图到卡牌上。需要指出的是，虽然游戏中有 3*3*3*3 共 81 中不同的卡牌，但是不同数目的卡牌可以通过多次贴图完成，不同颜色的卡牌可以通过图像重新着色完成，因此，需要准备的卡牌为 3*3 共 9 张即可。

关于图像纹理重新着色的细节位于:

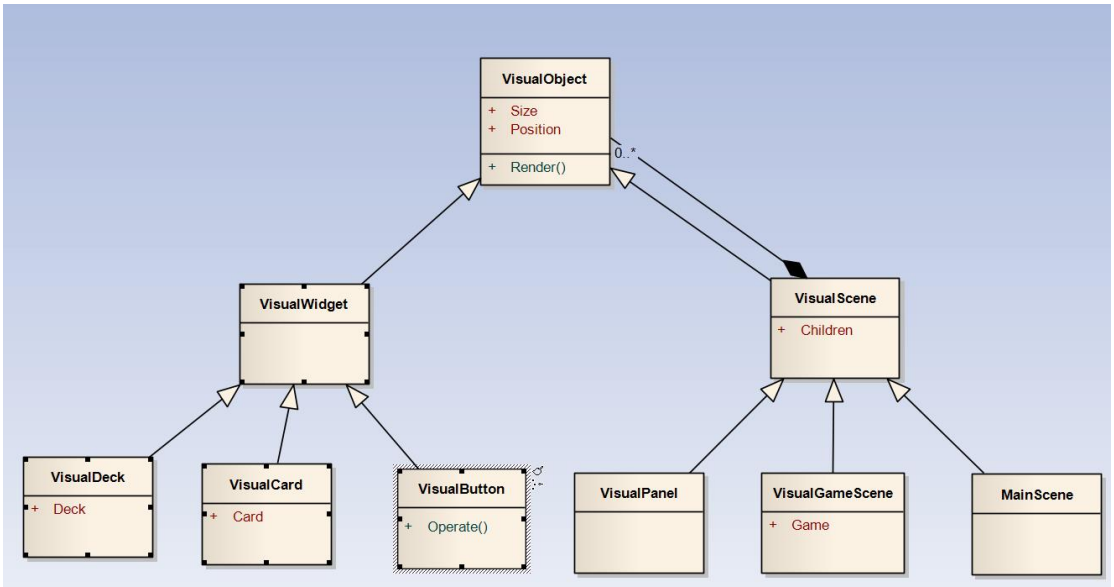
Include/Rendering/TextureManager.h

项目中使用的纹理文件及所有其他资源位于：

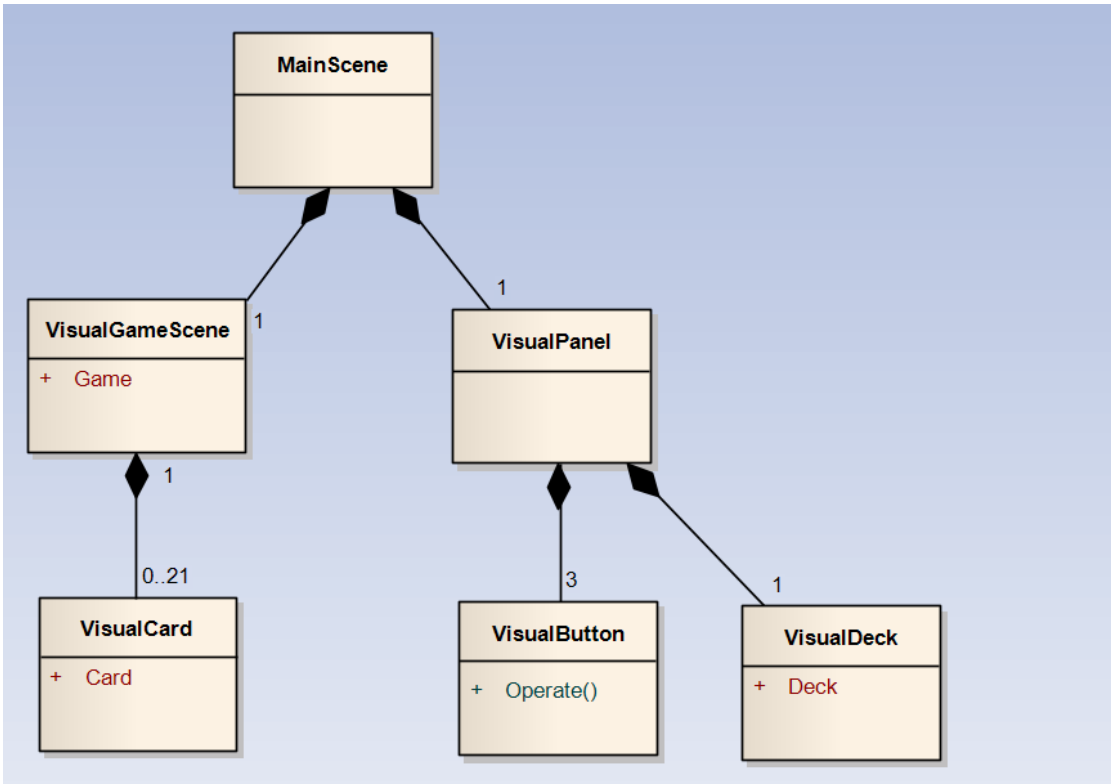
Res/

五、 OpenGL 显示框架

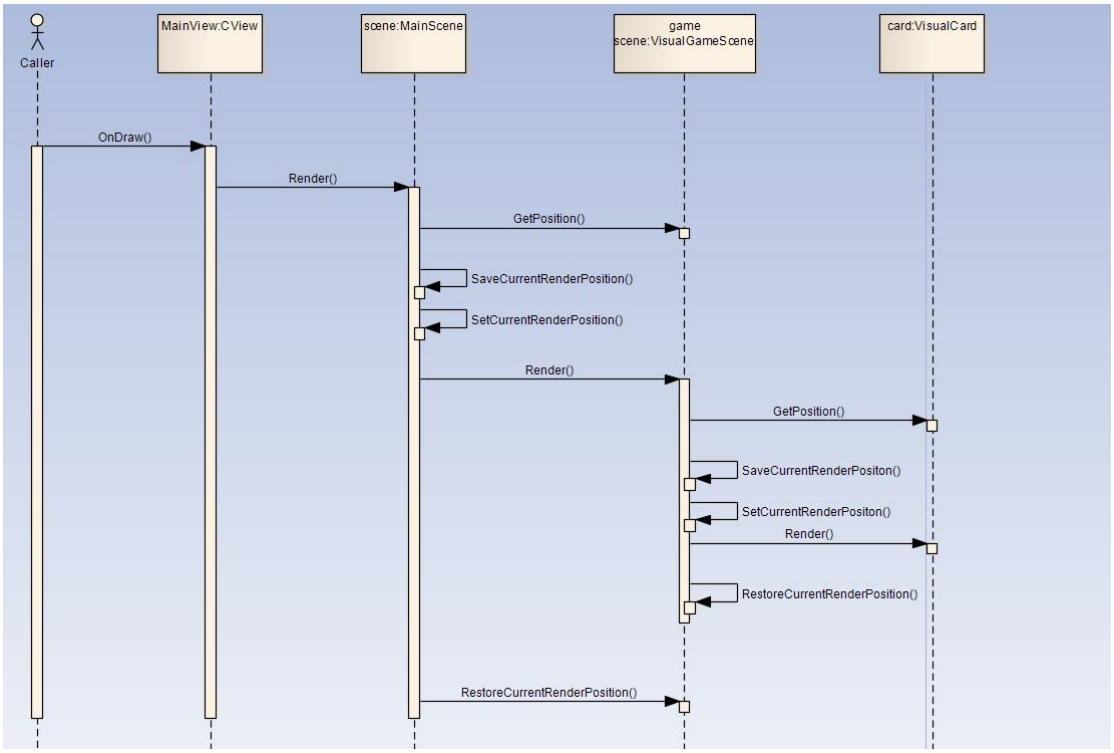
本项目使用 OpenGL 进行游戏渲染，为游戏中的物体如卡牌、按钮、面板等设计了专门的 3D 显示框架。相关类图如下：



整个结构利用组合设计模式。VisualObject 是场景中所有可见物体的抽象基类，所有的可见类均有自己的位置和大小，位置和大小均为三维度量，物体也可以渲染自己。VisualWidget 是一个鼠标可点击物体，为物体的鼠标点击时的通用视觉效果和涉及鼠标事件建模。VisualCard 和 VisualButton、VisualDeck 为具体的可见物体，分别为卡牌、按钮和牌堆。VisualScene 是 VisualObject 的集合，通过逐个渲染每个子物体来渲染自己。VisualPanel、VisualGameScene 和 MainScene 分别表示游戏控制面板、游戏主面板和整个游戏界面。上述各个类的包含关系如下：



下面的顺序图近似了渲染游戏场景中一个卡牌的过程：



为了方便控制，并提高可扩展性，本游戏并非采用事件驱动，而采用 Timer 驱动渲染，由主 Timer 定期发送渲染消息（20ms），刷新界面。

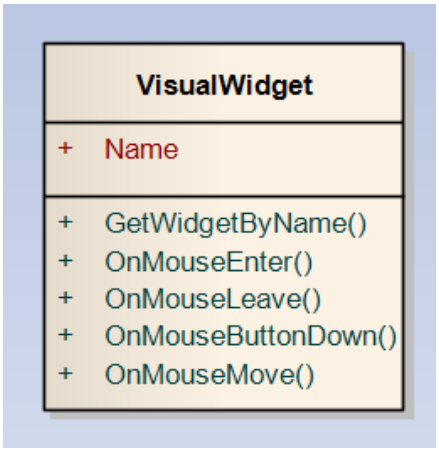
OpenGL 显示框架定义在：

Include/Rendering/

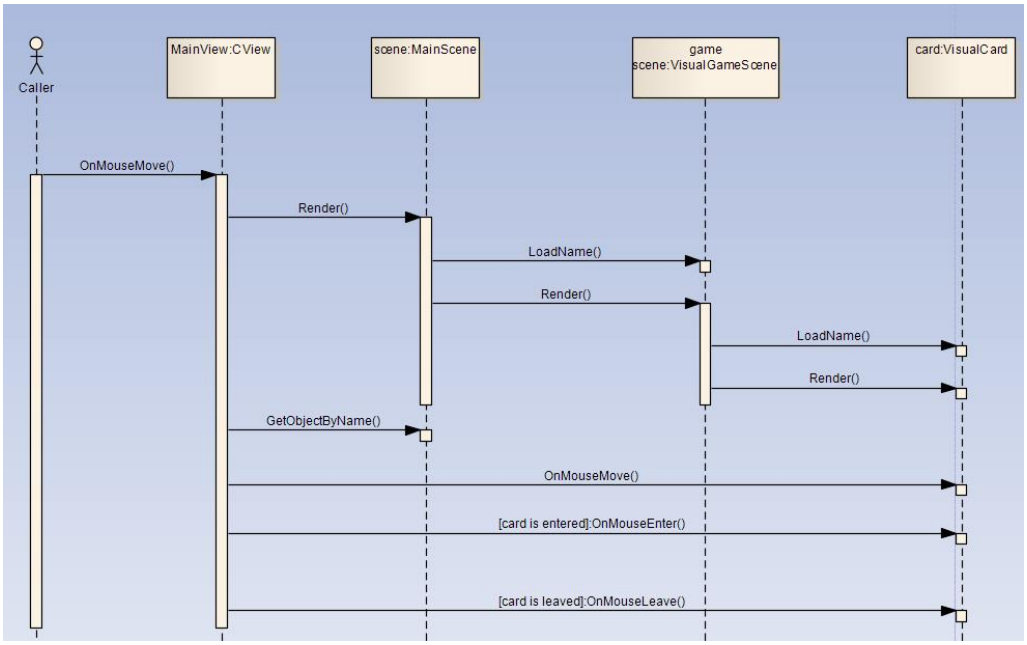
六、 OpenGL 点选设计

为了完成鼠标选择、悬停和点击的效果，本项目使用 OpenGL 内置点选功能来进行鼠标击中测试。为此，将 OpenGL 的点选功能集成到上述的 OpenGL 框架中。

具体来讲,为 VisualWidget 增加几个成员：



同时，将 OpenGL 加载名称的调用整合到渲染过程中，下面的顺序图简要说明了这一点：

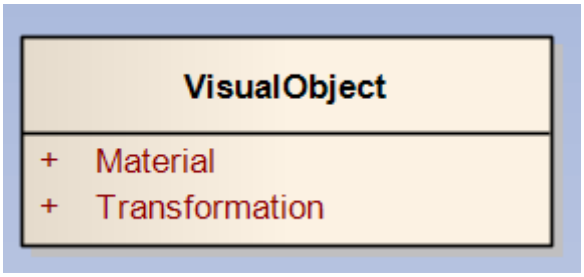


经测试，OpenGL 拾取在某些低端显卡上支持不够。因此，本项目另加入了一套基于屏幕坐标的 CPU 拾取框架，根据 OpenGL 渲染环境计算屏幕坐标，从而实现拾取功能。这里不再详述。

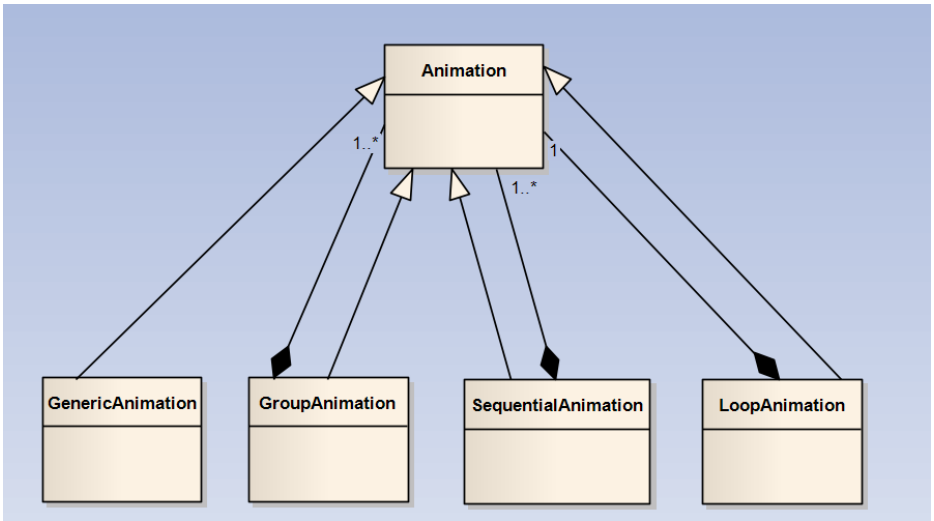
七、 动画框架设计

本项目为游戏中的效果设计了一个简单的动画框架。

首先，为 VisualObject 添加用于显示的若干属性：



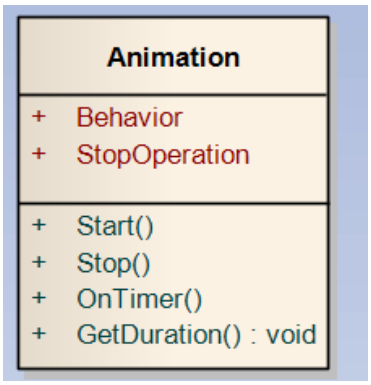
分别为物体的材质和物体的形变。程序中可以通过物体材质的变换、物体的平移、放缩、选择等形变来完成各式各样的动画效果。动画的类结构如下图：



Animation 为动画抽象基类，GenericAnimation 为泛型动画类，可以接受任何满足特定接口的对象作为其动画行为。项目的 Include/Animation/VisualObjectAnimations.h 中定义了一系列可以用来构造 GenericAnimation 的类型，有 Rotate、Transform、MoveVisualObject、Shake、Blink 等，这些对象通过改变 VisualObject 的特定属性来完成动画。

GroupAnimation 和 SequentialAnimation 为组合动画类（组合设计模式），分别可以同时执行若干动画或者顺序执行若干动画。

一个 Animation 有如下主要的接口：



动画可以开始、结束、获取持续时间，OnTimer()则在每一帧进行调用，进行相应的动画行为。Behavior 和 StopOperation 需要进一步说明。

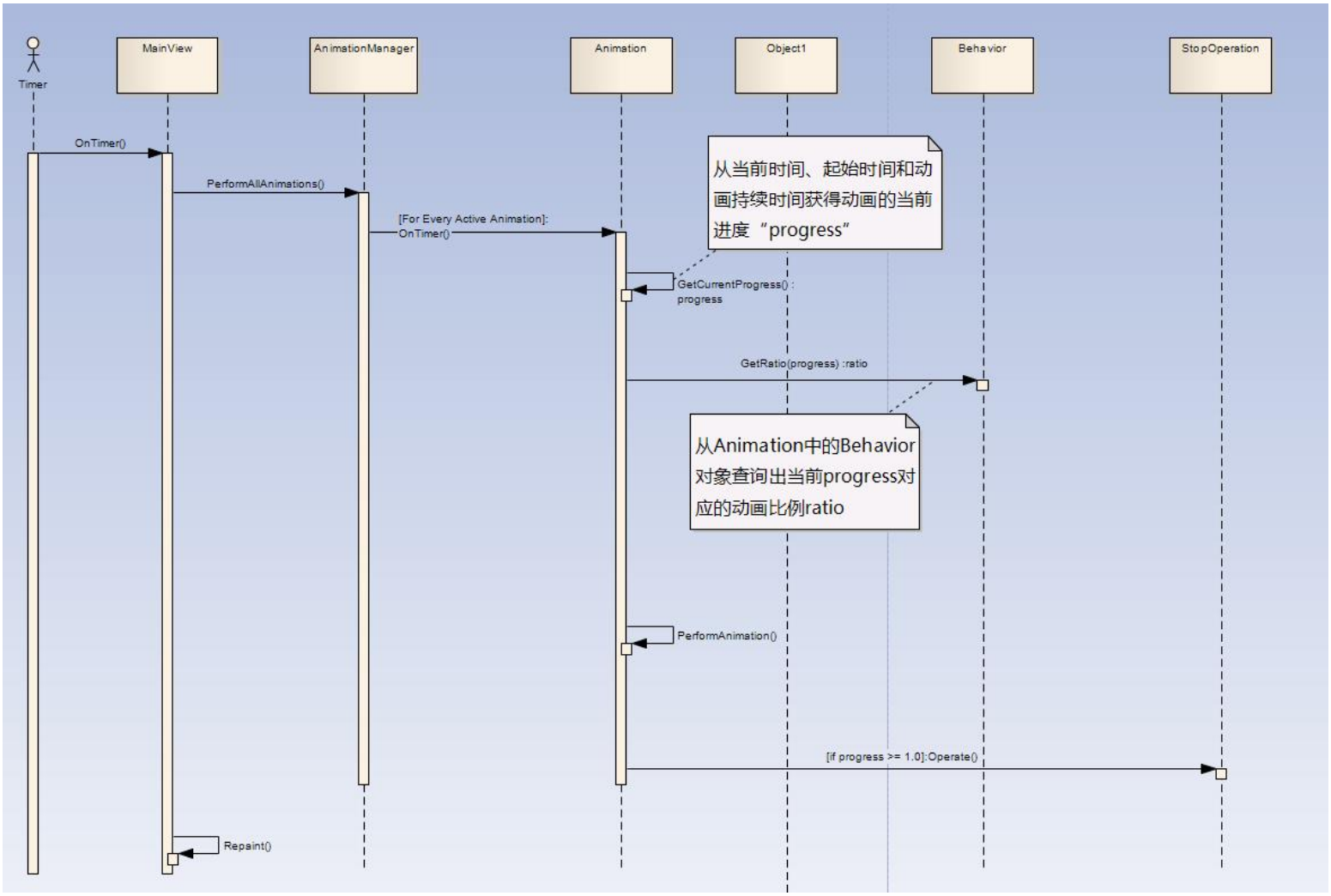
StopOperation 为一个结束行为，即在动画结束时调用的一段代码，可以接受任何可调用的函数子（函数、函数对象、Lambda 表达式等）作为行为。为了不在 Animation 中引入泛型参数，定义了 Operation 基类和 GenericOperation 泛型类，将泛型参数引入到 GenericOperations 中。

Behavior 定义了动画的运行行为（或者说轨迹），在动画运行的时候，动画从起始值到中止值不一定是线性变化的，如果需要完成复杂的动画曲线，如正弦、余弦或者震荡等动画运行的行为，可以通过定义特殊的 Behavior 来定制动画。同 StopOperation 一样，为了不再 Animation 中引入泛型参数，定义了 AnimationBehavior 和 GenericAnimationBehavior。

最后，为了让所有的动画能够统一运行并被管理起来，定义全局单例 AnimationManager，被启动的所有动画均自动注册到动画管理器，被停止的动画则从动画管理器注销。在每一个渲染周期，动画管理器负责调用所有注册的动画对象的

OnTimer 函数，以此修改 VisualObject 的相应属性。属性的变化会立刻在下一次渲染时显示出来，随着逐帧渲染，动画效果得以呈现。

一个的动画的运行过程由 Timer 驱动，其大致过程如下图所示：



动画框架定义在：

Include/Animation

八、 数据库、服务器设计

数据库使用 MySQL5.5，本项目定义两个表，分别是

USER	
ID	Integer, Primary Key
NAME	Char(20)

Score	
ID	Integer, Primary Key
USER_ID	Integer
SCORE	Integer
ELAPSED_TIME	Integer

USER 表记录用户名，SCORE 表记录用户 ID，分数和计时的记录。
服务器使用 Python 编写 CGI 程序，使用 Mysql-python 包连接数据库。

九、 杂项设计

另外程序还有一些杂项如下：
项目中使用的几何对象、秒表装置、随机数生成器等定义在：
Include/Utils/
项目中出牌带有音效，功能定义在：
Include/Sound/
项目中与服务器通信功能定义在：
Include/Web/