基于C++面向对象的飞机大战游戏开发

飞机大战游戏开发设计文档

版本 <2.0>

文档信息

|  |  |
| --- | --- |
| 标题： | 飞机大战游戏开发设计文档 |
| 文件位置： | :/517021910674/Document.docx |
| 版本： | <2.0> |
| 提交人： | 林江浩 |
| 提交日期： | 27/07/2018 |
| 状态： | 完成 |

修订历史记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** | **审核人** |
| 14/07/2018 | 1.0 | 第一次迭代文档撰写 | 林江浩 |  |
| 27/07/2018 | 2.0 | 第二次迭代文档撰写 | 林江浩 |  |
|  |  |  |  |  |
|  |  |  |  |  |

目录

1. 简介 4

1.1 目的 4

1.2 范围 4

1.3 定义、首字母缩写词和缩略语 4

1.4 参考资料 5

2. 软件架构 5

3. 静态设计 6

4. 动态设计 11

<飞机大战游戏设计文档>

# 简介

本项目在Windows 10 + Qt 5.8.0（包含Qt Creator 4.2.1）的开发环境下，基于C++语言，通过面向对象程序设计，开发一款PC端的飞机大战游戏。此文档意在阐明游戏的设计思路与过程，介绍相关的类与对象的抽象和设计；并为游戏玩家与其他使用该软件的用户提供指导与说明。

具体游戏规则、定义及缩略语说明、参考资料见下。

## 目的

此文档意在阐明游戏的设计思路与过程，介绍相关的类与对象的抽象和设计，并为游戏玩家与其他使用该软件的用户提供指导与说明。

## 范围

开发环境：Windows 10 + Qt 5.8.0（包含Qt Creator 4.2.1）

游戏规则：

1. 两种游戏模式：无限模式、胜利模式。
2. 无限模式中没有胜利条件，玩家不断追求高分，当玩家飞机与敌机发生碰撞时游戏结束。
3. 胜利模式中有胜利条件，当玩家得分达到目标分数时，玩家获得胜利。当玩家飞机与敌机发生碰撞时游戏结束。目标分数随游戏难度增加而增加。
4. 三种游戏难度：简单、困难、噩梦。随着难度的增加，敌机的出现速度、大型敌机出现概率、玩家子弹射速都会依次递增。
5. 简单模式中，敌机仅具有竖直速度，不会左右移动。
6. 困难模式中，敌机具有竖直速度和水平速度，移动方向和速度随机。
7. 噩梦模式中，敌机除了有竖直速度和水平速度，还具备一定的机器智能，可以优先躲避玩家子弹，并朝玩家飞机的方向移动
8. 玩家通过键盘方向键或者鼠标移动控制我方飞机移动，我方会自动发射子弹，射速、键盘移动灵敏度均可以调控。
9. 敌机出现频率可以调控，移动方向任意（噩梦模式除外），不会发射子弹，分为大、中、小三种，血量与体积依次递减。消灭小飞机需要1发子弹，得1000分；消灭中飞机需要5发子弹，得6000分；消灭大飞机需要20发子弹，得25000分。
10. 奖励道具随机出现，掉落频率可以调控，分为两种：全屏炸弹和双倍子弹。全屏炸弹为一次性道具，由鼠标右键触发，玩家在游戏开始时拥有五枚炸弹。双倍子弹为即时道具，获得后立即触发，持续时间随机。
11. 键盘P键可以暂停游戏，再按下P键可以继续游戏。
12. 游戏提供音乐开关，玩家在游戏暂停状态下可以选择打开或关闭音乐。
13. 游戏提供存档功能，玩家在游戏暂停状态下可以选择保存当前游戏状态。而后可以在游戏开始界面通过“读取存档”按钮读取以前保存的游戏状态。
14. 游戏提供记录与回放功能，玩家可在游戏结束界面选择保存该局游戏记录。而后在游戏开始界面可以通过“精彩回放”按钮播放以前保存的游戏记录
15. 游戏结束后会出现“重新开始”“结束游戏”“保存游戏”“返回菜单”四个选项。选择“重新开始”会重新开始一局新的飞机大战游戏，选择“结束游戏”则会关闭游戏窗口，选择“保存游戏”则会保存该局游戏的记录文件，选择“返回菜单”则可以返回游戏开始界面。

游戏扩展功能说明：

1. 提供多种游戏模式：无限模式和胜利模式
2. 提供多种游戏难度：简单、困难、噩梦
3. 可以记录玩家的历史最高分数，并在一局游戏结束后更新并显示
4. 提供音乐音效开关，玩家可以选择开启或关闭
5. 扩展玩家操作方式，同时支持键盘操作与鼠标操作
6. 机器语言评价：在一局游戏后根据玩家得分、目标分数和历史最高分数，给出不同的游戏评价，如“真遗憾，就差一点就胜利啦！”“革命尚未成功，同志还需努力！”

## 定义、首字母缩写词和缩略语

以下为游戏设计与文档设计过程中涉及的缩写前缀或单词的含义（仅针对特别的写词做出必要解释，不包含全部）：

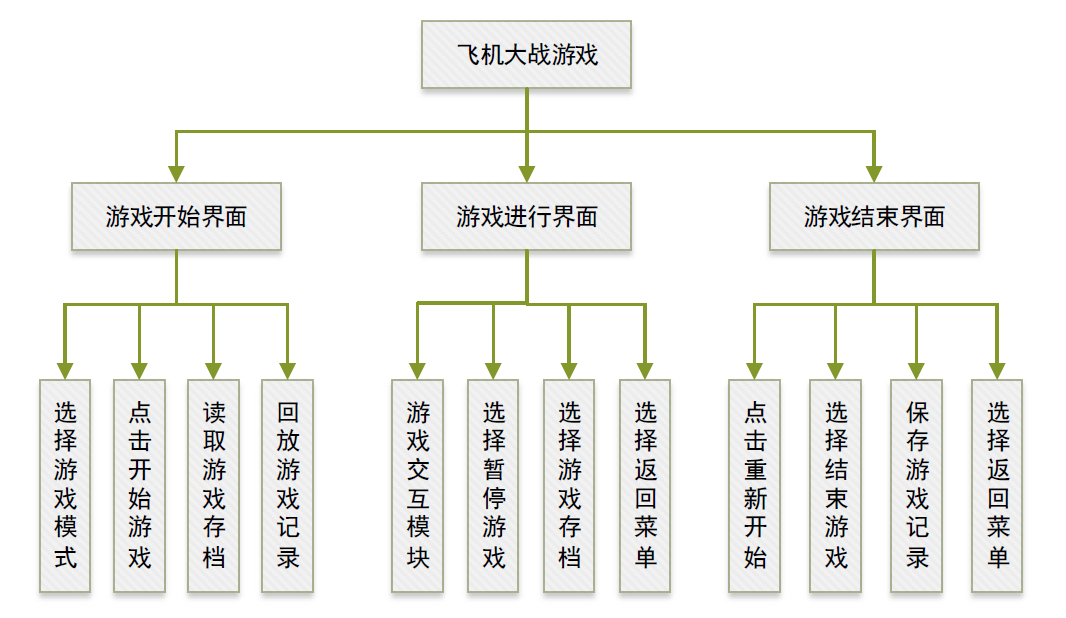
|  |  |
| --- | --- |
| 缩写前缀或单词 | 含义 |
| Gene- / Gen- | 产生，如 GenPlane() |
| Big | 表示敌机种类：大型敌机 |
| Mid | 表示敌机种类：中型敌机 |
| Small | 表示敌机种类：小型敌机 |
| -Down | 表示被消灭，如is\_Plane\_Down |
| t- / Time- | 表示与计时器相关的变量或函数 |
| get- | 表示获取某一变量值的函数 |
| set- | 表示修改某一变量值的函数 |
| Reward | 道具 |
| Review | 游戏回放 |

## 参考资料

|  |  |
| --- | --- |
| 文件名和位置 | 标题 |
| http://www.qter.org/portal.php?mod=view&aid=26 | Qt 快速入门系列教程 |
| :/517021910674/ Qt Creator快速入门(第三版).pdf | 《Qt Creator快速入门(第三版)》 |

# 软件架构

软件的功能分解如下图所示：



**图2.1 软件功能分解图**

游戏整体设计分为三个界面：开始界面、进行界面、结束界面。

游戏开始界面（也称“菜单”）可以提供三个按钮选项：开始游戏、读取存档、精彩回放。其次还有两个单选按钮，可供玩家自由选择游戏模式和游戏难度。

游戏进行界面首先是有一个持续的游戏交互模块（即游戏过程），其次还提供暂停游戏选项（P键），玩家在暂停游戏后可以选择存档游戏、返回菜单以及音乐开关。

在一局游戏结束后，跳转至游戏结束界面，会显示玩家的该局最终得分和历史最高分数，同时根据情况给出不同的机器评价（如“恭喜你，获得胜利！”“唉，就差一点就成功了！”）。游戏结束界面下方提供四个按钮：重新开始、结束游戏、保存游戏（即保存该局游戏记录用于回放）、返回菜单。

# 静态设计

首先，游戏中有玩家飞机（Plane）、敌机（Enemy）、道具（Reward）、子弹（Bullet）四个类，而这四个类都属于实体，具有坐标、移动速度、长、宽、图像等基本元素，故可抽象出一个基类：实体类（Entity）。其次，游戏需要音效与飞机被消灭时的动画效果，因此设计了爆炸效果类（Explosion）和音乐播放类（MusicPlay）。

在此基础上设计一个工厂类（Factory），用以构造并返回以上各类的对象。

为了配合界面绘制，抽象设计了一个继承自Qt基类QPushbutton的游戏按钮类（MyButton）。

设计一个游戏回放类（Review），继承自Qt的基类QWidget。

另外实现游戏逻辑与GUI的分离，设计游戏逻辑类（Game\_Logic）和GUI类（gui\_app）。游戏逻辑类保存以上各类的对象的参数并对其进行操作；GUI类继承自Qt的基类QWidget，获得事件，调用游戏逻辑中对应函数进行操作，并根据返回值绘制界面。

综上，飞机大战共设计12个类，下面先给出各个类的框图，后用UML类图表示其相互关系。



图3.1 Class Entity



图3.2 Class Plane（继承自Class Entity）



图3.3 Class Enemy（继承自Class Entity）



图3.4 Class Bullet（继承自Class Entity）



图3.5 Class Reward（继承自Class Entity）



图3.6 Class Factory



图3.7 Class MusicPlay



图3.8 Class Explosion



图3.9 Class MyButton（继承自QPushButton）



图3.10 Class Review（继承自QWidget）



图3.11 Class gui\_app（继承自QWidget）



图3.12 Class Game\_Logic

各类之间的关系，以UML类图表示如下，为了表达简明，选择性的删去了一些类中的成员，类内部的具体成员信息请参见上述的框图：

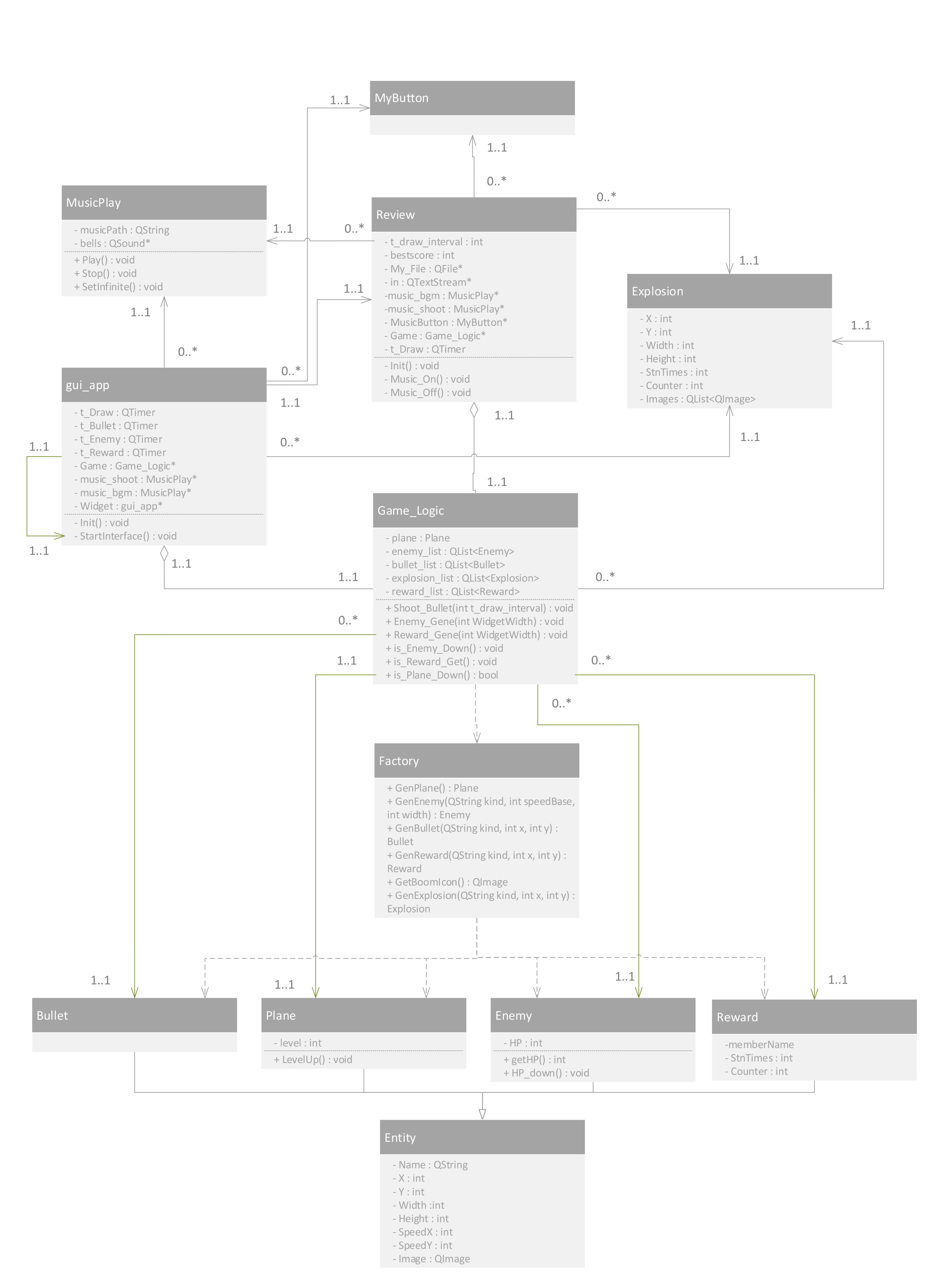


图3.13 UML Class Diagram

# 动态设计

# 4.1 综述

玩家的游戏与操作均在窗口界面MyWidget（class gui\_app）上进行。在gui\_app类的对象MyWidget中定义一个Game\_Logic类的对象Game。MyWidget获得事件后，调用Game中对应的逻辑函数来处理该事件，逻辑函数处理完后返回一个参数，指导MyWidget对可视化窗口界面做出相应的修改绘制，从而达到玩家与程序交互的目的。

MyWidget中可能产生的事件有：鼠标移动事件（MouseMove），鼠标单击事件（MousePress），键盘事件（KeyPress），绘图计时器到时事件（t\_Draw\_Timeout），发射子弹计时器到时事件（t\_Bullet\_Timeout），掉落道具计时器到时事件（t\_Reward\_Timeout），产生敌机计时器到时事件（t\_Enemy\_Timeout）。除此之外，还有玩家选择“精彩回放”功能，则会触发回放事件（Review）。

因为整体程序设计的UML顺序图过于冗长，故按照事件归类，针对不同的事件用不同的UML顺序图体现动态交互，具体如下：

# 4.2 鼠标与键盘事件

当玩家通过鼠标、键盘和游戏进行交互时，由界面MyWidget接收事件，然后调用游戏逻辑Game中对应的函数对Plane、Enemy、Bullet、Reward与其他参数进行修改，同时进行各类碰撞检测，然后返回参数值指导MyWidget 的修改。

但是鼠标和键盘事件本身不修改可视化界面，即被修改的对象与参数还不会体现在可视化界面中。

具体UML顺序图如下：

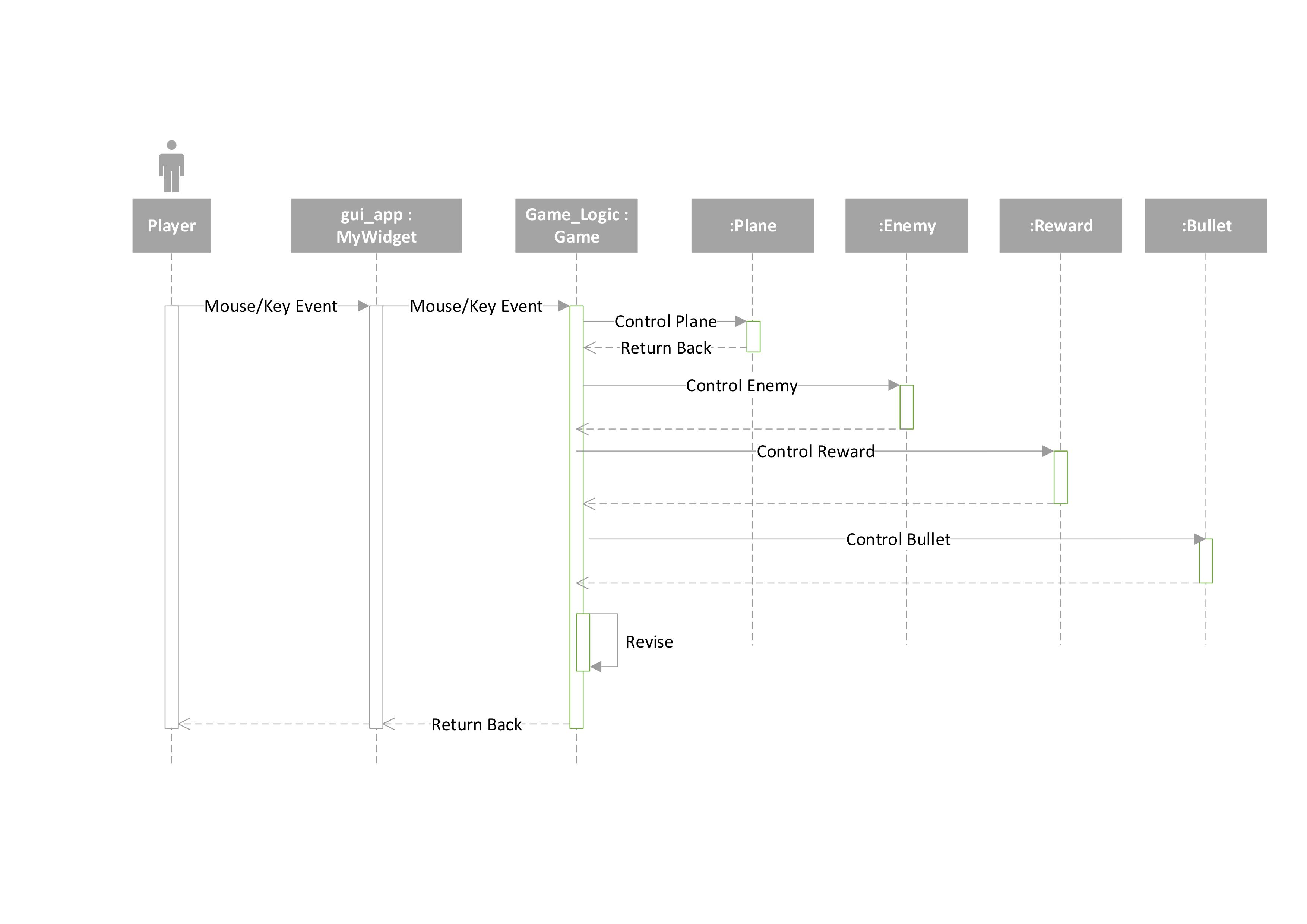


图4.2.1 鼠标键盘事件的UML顺序图

# 4.3 发射子弹计时器到时事件

t\_Bullet计时器到时后，MyWidget调用Game中ShootBullet( )函数，根据双弹时间向子弹列表bullet\_list中添加子弹。同样的，此时对对象与参数的修改不会在可视化界面中体现。

具体UML顺序图如下：

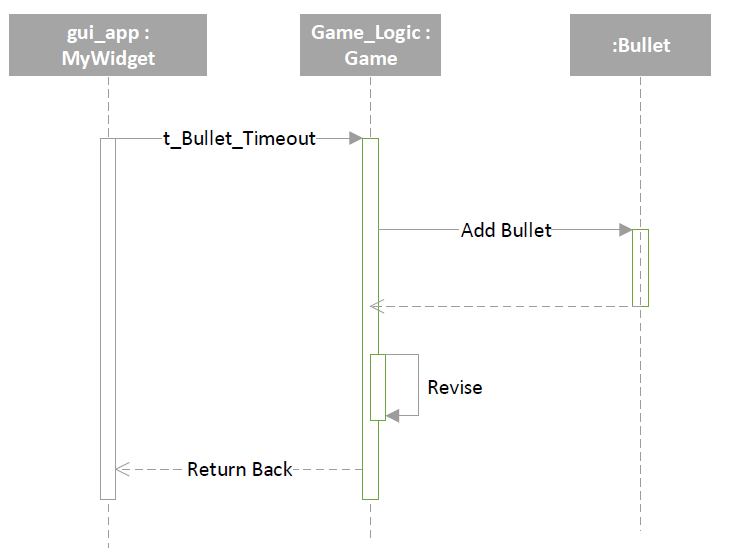


图4.3.1 发射子弹计时器到时事件UML顺序图

# 4.4 掉落道具计时器到时事件

t\_Reward计时器到时后，MyWidget调用Game中GeneReward( )函数，通过随机函数向道具列表rewar\_list中添加道具。同样的，此时对对象与参数的修改不会在可视化界面中体现。

具体UML顺序图如下：

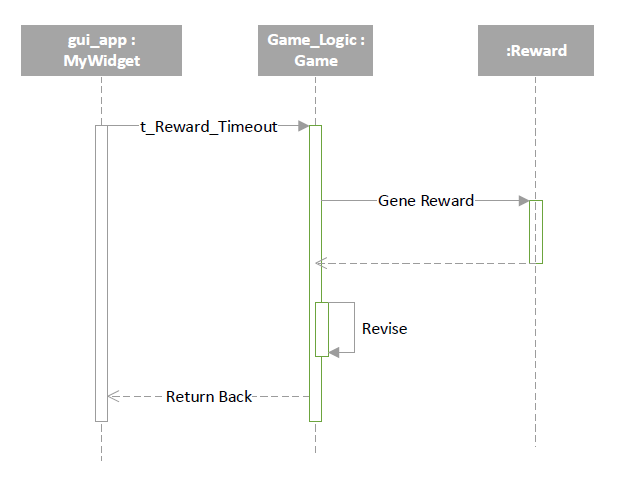


图4.4.1 掉落道具计时器到时事件UML顺序图

# 4.5 产生敌机计时器到时事件

t\_Enemy计时器到时后，MyWidget调用Game中GeneEnemy( )函数，通过随机函数向敌机列表enemy\_list添加敌机。同样的，此时对对象与参数的修改不会在可视化界面中体现。

具体UML顺序图如下：

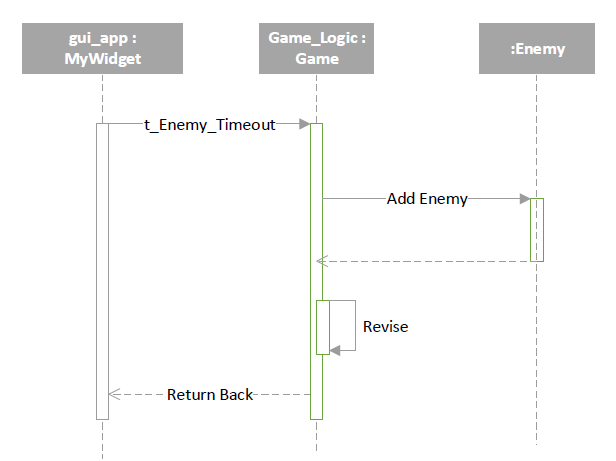


图4.5.1 产生敌机计时器到时事件UML顺序图

# 4.6 绘图计时器到时事件

t\_Draw计时器到时后，MyWidget调用自身的PaintEvent( )函数对可视化窗口进行重新绘图，此在这个绘图周期中所有的对象与参数值的修改通过绘图在窗口上体现，实现与玩家的交互。

具体UML顺序图如下：

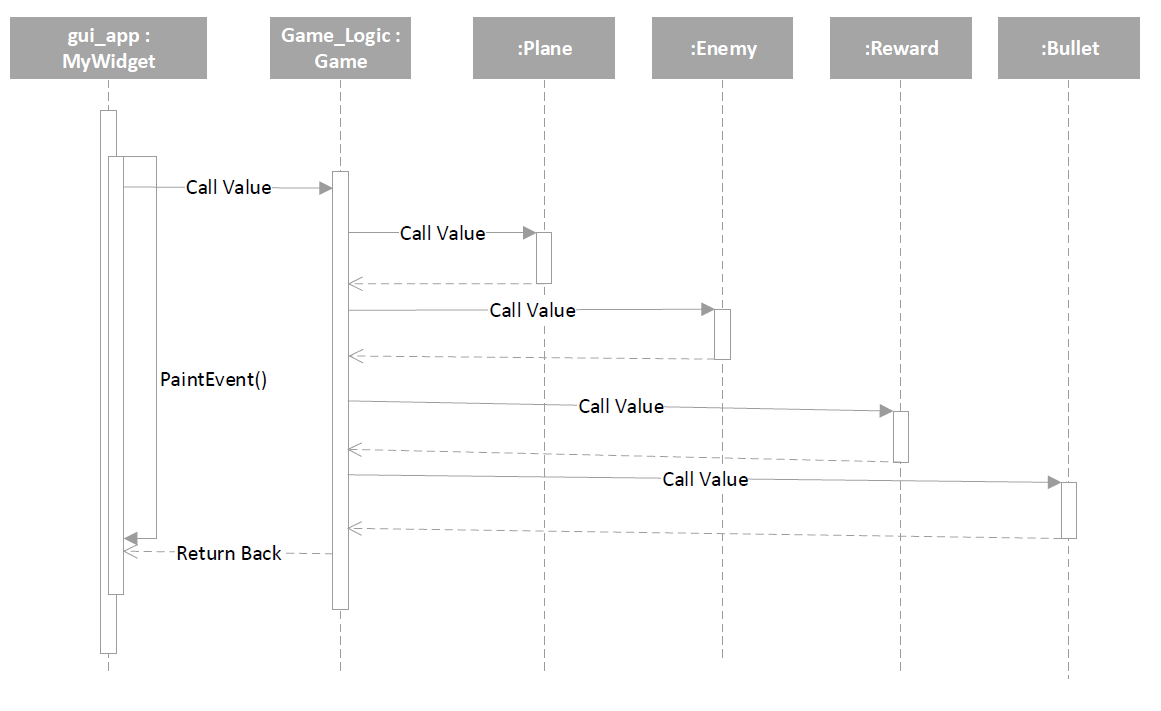


图4.6.1 绘图计时器到时事件UML顺序图

# 4.7 回放事件

玩家选择“精彩回放”功能，在选择一个游戏记录文件后，MyWidget将调用Review类，弹出一个新的播放窗口Widget，游戏窗口MyWidget和回放窗口Widget可以同时工作。

具体UML顺序图如下：

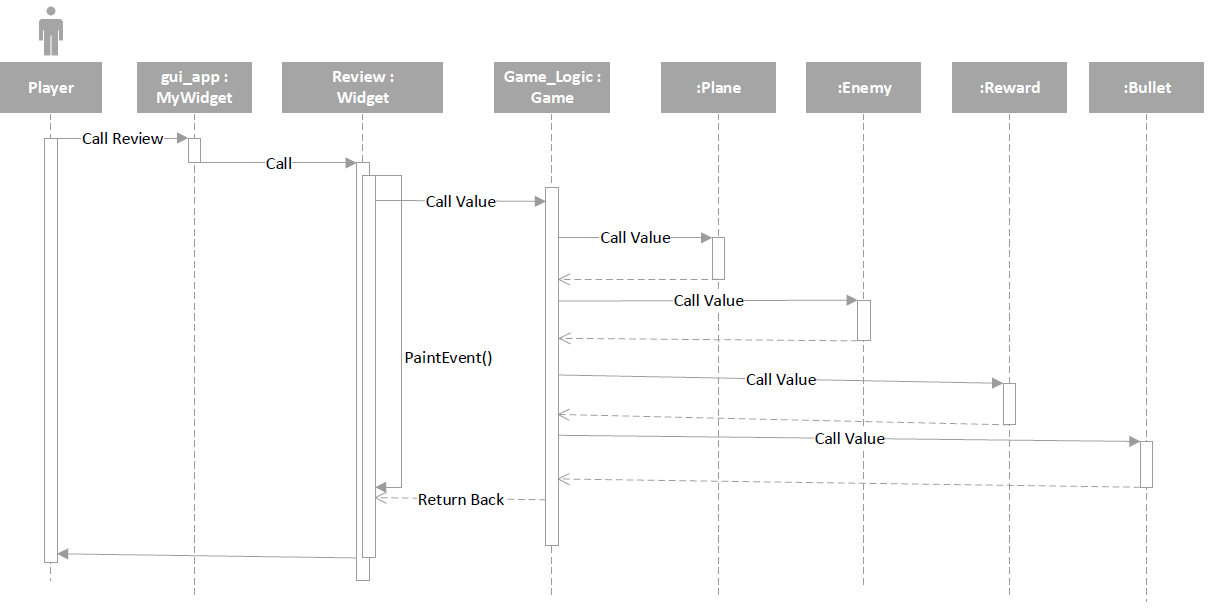


图4.7.1 回放事件顺序图