大美 **mixer**　简书>>https://www.jianshu.com/u/84f0ebbbac87
Github>>https://github.com/lmh760008522　**CSDN**>>https://blog.csdn.net/qq_34746896

南京邮电大学物联网学院

2016 / 2017 学年第 1 学期

《JAVA 程序设计》课程大作业报告

课题代号（单选）☑课题 1　　☐课题 2　　☐课题 3

课题名称：　炮台打飞机

小组成员人数（单选）☐1 人　　☐2 人　　☑3 人　　☐4 人　☐5 人

小组成员名单(学号和姓名）：

(1)学号　B15070204　姓名　刘美含

(2)学号　B15070312　姓名　薛睿

(3)学号　B15070310　姓名　钟铭

课题实现采用的开发工具（如为其他，请填写具体工具名称）
☐NetBeans ☑Eclipse　☐JDK 命令行 ☐其他_____

课题实现采用的操作系统（如为其他，请填写操作系统名称）
☑Microsoft Windows　☐Unix　☐ Linux ☐Mac OS　☐其他_____

课题实现形式（如为 PC 桌面应用程序之外的其他形式请填写）_____

课题完成时间：　2016 年 12 月 20 日

对应提交的电子文档文件夹名称(准确填写)

B15070204_B15070312_B15070310_1

==========以下部分为教师填写区，请勿填写==========

评阅编号:（　　）—（　　）

成绩评定：

软件运行_____软件基本功能_____软件提高功能_____　软件部分成绩_____

文档结构_____文档理论_____文档撰写_____　文档部分成绩_____

总成绩_____

备注_____

目　　录

一、软件说明和亮点分析

1、软件说明

（1）软件名称：**飞机大战**

（2）运行环境：windows 系统

开发环境：eclipse

开发语言：java\html

（3）玩法：

炮台打飞机

地面有一座炮台，可以沿水平方向移动。天空中敌机随机飞过，可向下投掷炸弹。炸弹击中炮台后炮台将受损，击中三次则游戏结束。玩家可用鼠标移动控制炮台向上发射炮弹击中飞机。游戏过程中会出现下落的降落伞。若接住，则玩家炮弹将出现升级奖励。小型敌机，击中 1 次即落，得分 1000 分。中型敌机，击中 2 次即落，得分 2000 分。

大型敌机，击中 3 次即落，得分 3000 分。

飞机打飞机

与炮台打飞机玩法相近。炮台变成可自由移动的飞机。玩家用鼠标移动控制飞机向敌机发射炮弹。我方飞机与敌机相撞之时，游戏结束。ps.游戏过程中显示得分，结束后显示总分。

2、亮点分析

（1）操作简单、方法易学：用户只需动动鼠标就能发现游戏的窍门，无需复杂的新手教学或键盘操作，适用人群广。

（2）界面美观、音效出色：我们适用风和简约但是加入了赏心悦目的背景，使用户在体验简约美的同时欣赏星空之美，营造出了良好的意境。经典音效的应用让人仿佛穿越时代，复古风格十足。

（3）简单却吸引人：游戏玩法十分简单，但是不容易让人产生厌倦的秘诀是每个游戏只设置一个关卡但是逐渐增加难度，增加挑战性。排行榜的使用让用户不停的想刷新记录，从而不放弃游戏。永远不可能通关的原则利用了人的“自圆心理”，自然不停的玩游戏。

（4）点开“帮助”，会跳出一个精美的 HTML 页面，一方面图形与文字结合，清晰明了的展示了游戏玩法，另一方面可以找到创作者的联系方式，与创作者进行对话。

二、软件设计方案

1、功能设计

炮台打飞机有以下的功能：

（1）地面有一座炮台（我方飞机），可以沿水平方向移动。

（2）天空中有自由出现的敌机以某个角度从炮台上方飞过。

（3）中型和大型飞机可以向下投掷炸弹，炸弹击中炮台后炮台生命值减少一，炮台被击中三次后将炸毁，游戏结束。游戏过程显示生

命值。

（4）炮台可以垂直向上发射子弹，炮弹击中飞机就得分，飞机一共有三种，大型飞机需要击中二十次才会爆炸，中型飞机需要击中飞机十次，小型飞机则需击中一次就会爆炸，爆炸消失后重复上述游戏过程。

（5）会出现落下的蓝色降落伞，若接住，则出现子弹升级奖励。子弹升级后发射出两排蓝色子弹，奖励时间结束后，恢复普通炮弹。

（6）游戏过程中显示游戏得分，击中小型飞机 1000分，
击中中型2000分，击中大型3000分，游戏结束后显示游戏总分。

（7）考虑到游戏可玩性，先出现小飞机，慢慢出现中型飞机，大飞机出现的频率最低，所需要的炮弹也最多，但是得到的分数也最多。飞机总体出现的频率越来越短，用户所面对的难度越来越大，越来越具有挑战性。

飞机打飞机实现了以下功能：

（1）画面中有一架飞机（我方飞机），可以随鼠标自由移动。

（2）天空中有多个敌机从不同位置向我方飞机飞来。

（3）我方飞机与敌机相撞之时，游戏结束。

（4）飞机可以垂直向上发射子弹，炮弹击中敌机则加分。敌机一共有三种，大型需要击中二十次会爆炸，中型飞机需要击中飞机十次，小型飞机需要击中一次会爆炸，爆炸消失后重复上述游戏过程。

（5）会出现落下的蓝色降落伞，若接住，则出现子弹升级奖励。子弹升级后发射出两排蓝色子弹，奖励时间结束后，恢复普通炮弹。

（6）游戏过程中显示游戏得分，击中小型敌机 1000分，
击中中型6000分，击中大型30000分，游戏结束后显示游戏总分。

（7）考虑到游戏可玩性，先出现小飞机，慢慢出现中型飞机，大飞机出现的频率最低，所需要的炮弹也最多，但是得到的分数也最多。飞机总体出现的频率越来越短，用户所面对的难度越来越大，越来越具有挑战性。

2、软件总体设计

（1）游戏程序是一项精度要求很高的程序系统，因为其代码利用率很高。一个实时运行的最终作品，每秒都会运行成千上万行程序，绘图事件、键盘事件都会以极高的频率在后台等待响应，若有丝毫的差别都将很容易导致程序在运行不久后可能出现严重错误，甚至死循环。因此，其逻辑设计应当相当严谨，需将所有可能发生的事件及意外情况考虑在设计中。

（2）游戏中为了美观，适用性强，可能需要采用外部文件引入的图片贴图，我们都有较好的解决方案。

（3）玩家飞机的运行可以通过键盘响应事件控制，但敌方则因为是自动运行，就需要有一定的智能性；敌人飞机的运行算法也要进行

相关的设置，已免游戏过于简单，增加了游戏的可玩性。

（4）双方的飞机在前进时也需要考虑到是否碰撞到对方飞机，以免重叠运行，造成许多物理上不可能的情况，缺乏真实感。每一次刷新页面、每前进一步都需要进行相关的碰撞检测。

（5）游戏的图片不可能通过绘图来解决。否则，不仅难于控制和处理过多的元素，也会因过多的大型图片而不能限制程序的大小，失去程序的原则和 Java 的优势。

（6）Java 是基于虚拟机的半解释型编译系统，其执行效率较 C++等完全编译后的程序会低很多，程序如果不进行精简和优化，将可能导致运行的不流畅。除开发过程中对结构上的控制、变量的使用、算法的优化等优化外，还可以使用混淆器(Obfuscator)进行程序打包后的优化。

（7）游戏的结束、开始、动态信息画面作为构成一个程序都是必不可少的重要部分。良好的用户界面更是吸引用户的硬指标，相关的美术构图和人性化设置也需要有一定的考虑。

三、软件系统分析和算法分析

1、子弹位置变化函数

```
public void onBulletLocationChanged(Bullet b) {
if (b != null) {
    b.setPosY(b.getPosY() - b.getSpeed());//使子弹不停的移动
    if (b.getPosY() <= 0) {//子弹飞出屏幕
  synchronized (this.bullets) {
     this.bullets.remove(b);
  }
    }

    EnemyPlane enemyPlane = b.hitEnemyPlanes();
if (enemyPlane != null) {//若没打到
  enemyPlane.drawFighting(this.getComponentGraphics(this.getGraphics()));
  if (enemyPlane.isKilled()) {//若打到
        //根据打到敌军飞机的不同播放不同的音乐
     switch (enemyPlane.getEnemyType()) {
     case SMALL_ENEMY_PLANE:
        this.smallPlaneKilledSoundPlayer.play();
        break;
     case BIG_ENEMY_PLANE:
        this.bigPlaneKilledSoundPlayer.play();
        break;
     case BOSS_ENEMY_PLANE:
        this.bossPlaneFlyingSoundPlayer.stop();
        this.bossPlaneKilledSoundPlayer.play();
        break;
     }

     //加分
     synchronized (this) {
        this.score += enemyPlane.getKilledScore();
     }
     //移除敌军飞机
     synchronized (this.enemyPlanes) {
        this.enemyPlanes.remove(enemyPlane);
```

```
        }

        //移除子弹
        synchronized (this.bullets) {
            this.bullets.remove(b);
        }
        enemyPlane.drawKilled(this.getComponentGraphics(this.getGraphics()));
    }
     }
}
}
```

（1）先根据子弹速度改变子弹位置。

（2）判断是否撞到敌机。

（3）若撞到，播放音乐，增加分数，绘制爆破，移除子弹。

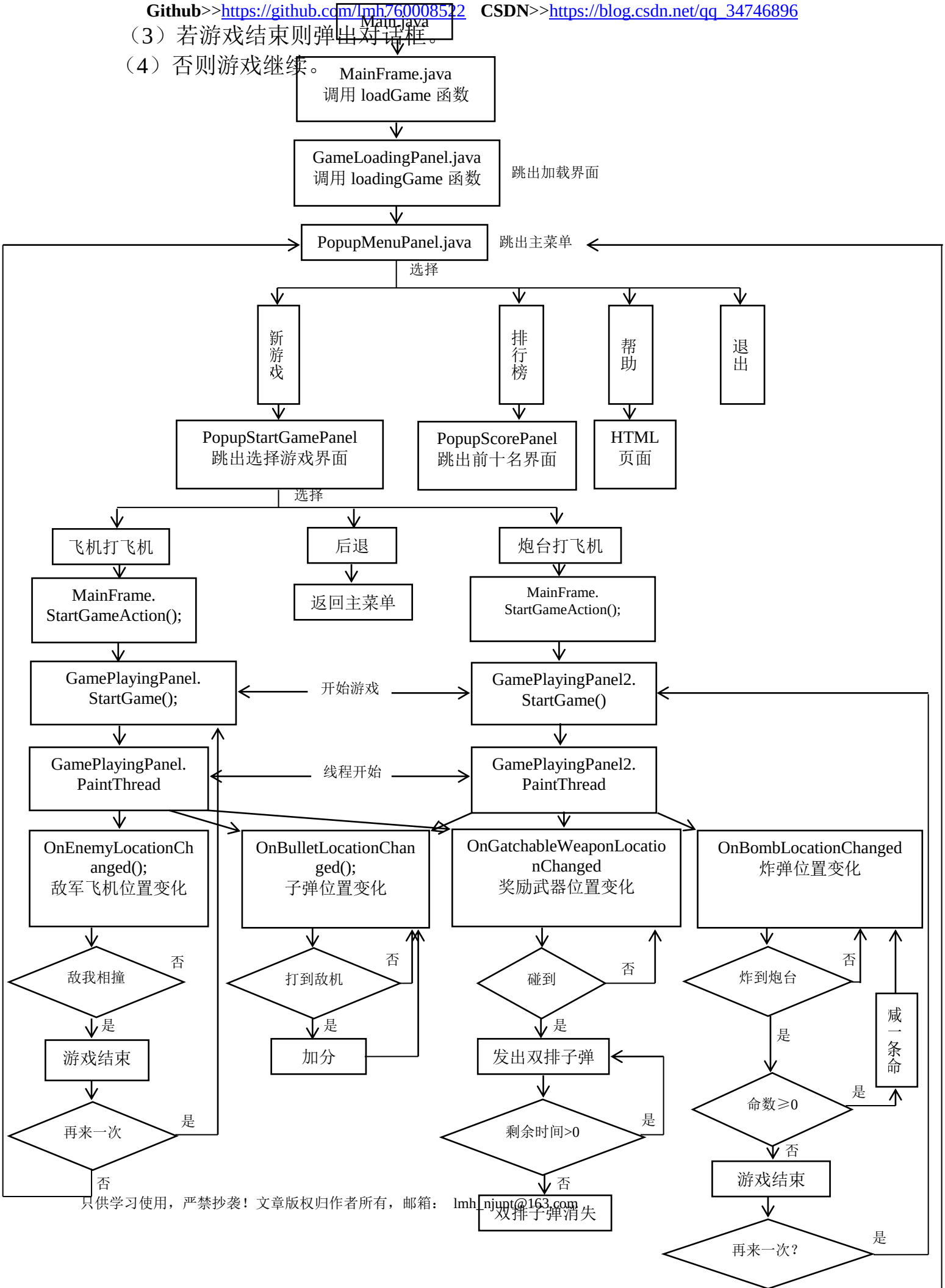
2、飞机打飞机开始游戏函数

```
    private void startGame() throws LineUnavailableException,
UnsupportedAudioFileException, IOException {
        Container c = this.getContentPane();
        c.removeAll();
        this.repaint();
        BorderLayout borderLayout = new BorderLayout();
        c.setLayout(borderLayout);
        this.gamePlayingPanel = new GamePlayingPanel();
        c.add(this.gamePlayingPanel, BorderLayout.CENTER);
        this.gamePlayingPanel.startGame();
        long startTime = System.currentTimeMillis();//记录开始时间
        while (this.gamePlayingPanel.getMyPlane().isAlive()) {
            try {
                Thread.sleep(Config.GAME_PANEL_REPAINT_INTERVAL);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
        long endTime = System.currentTimeMillis();//记录结束时间
        int option;
              if(this.addScore(this.gamePlayingPanel.getScore(), endTime -
startTime)==1){
              option = JOptionPane.showConfirmDialog(this, "新纪录: " +
this.gamePlayingPanel.getScore()
                                    + "! ! ! 再来一次？","游戏结束",
JOptionPane.YES_NO_OPTION);
              }else{
              option = JOptionPane.showConfirmDialog(this, "失败！分数: " +
this.gamePlayingPanel.getScore()
                              + "。再来一次？", "游戏结束",
JOptionPane.YES_NO_OPTION);//结束弹出选择框
              }
              switch (option) {
              case JOptionPane.YES_OPTION:
              startGame();
            break;
              case JOptionPane.NO_OPTION:
            stopGame();
            break;
        }
       }
```

（1）先绘制版面。

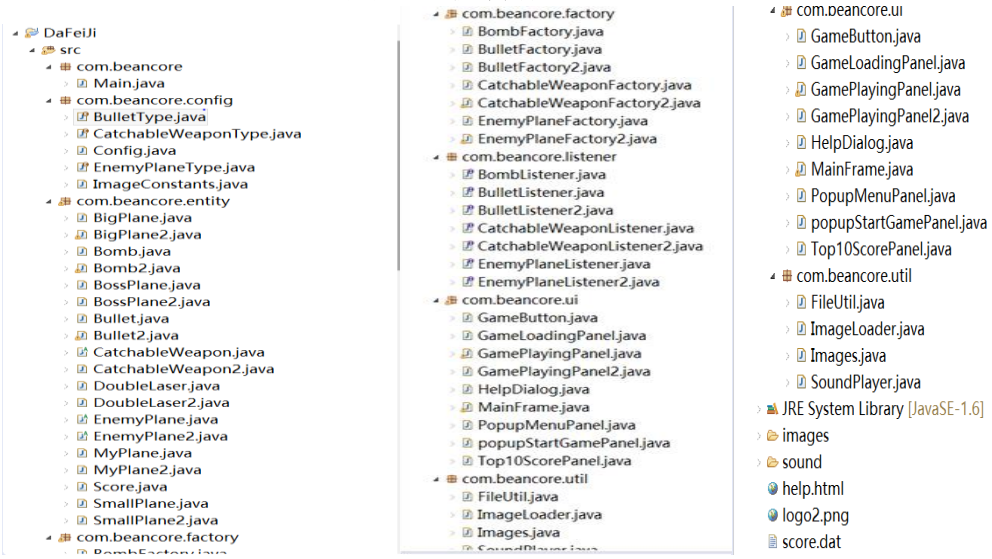
（2）开始游戏。

（3）若游戏结束则弹出对话框。
（4）否则游戏继续。



四、软件实现和代码编写

下图为软件所用的包



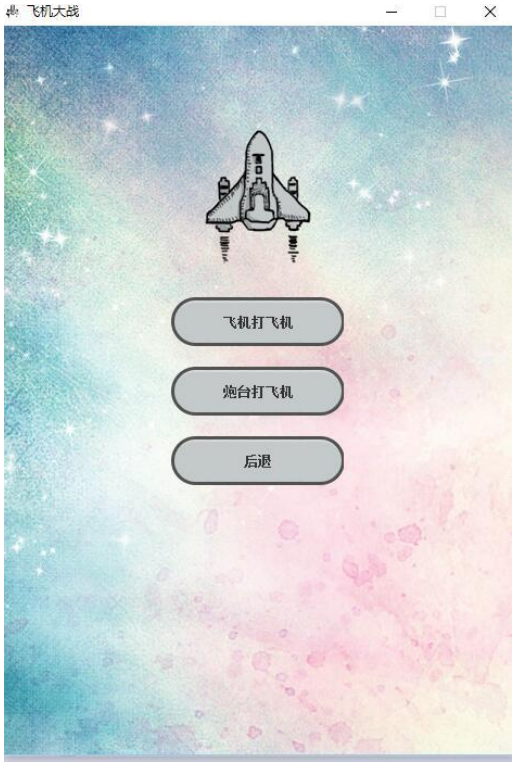
包和文件夹说明表	
主要包和文件夹名称	包和文件夹的说明
com.beancore	存放 MAIN 函数
com.beancore.config	游戏各种定义和固定的参数和命名
com.beancore.entity	单个组件的功能设定
com.beancore.factory	飞机、子弹、炸弹的产生函数
com.beancore.listener	存放各种监听器
com.beancore.ui	存放所有界面函数
images\sound 文件夹	存放游戏所需要的文件和音频

主要代码见附录。

五、软件的运行和测试

图一：软件主界面

图二：点击新游戏后的界面



图三：点击飞机打飞机后的界面

图四：获得奖励和奖励作用时间的界面

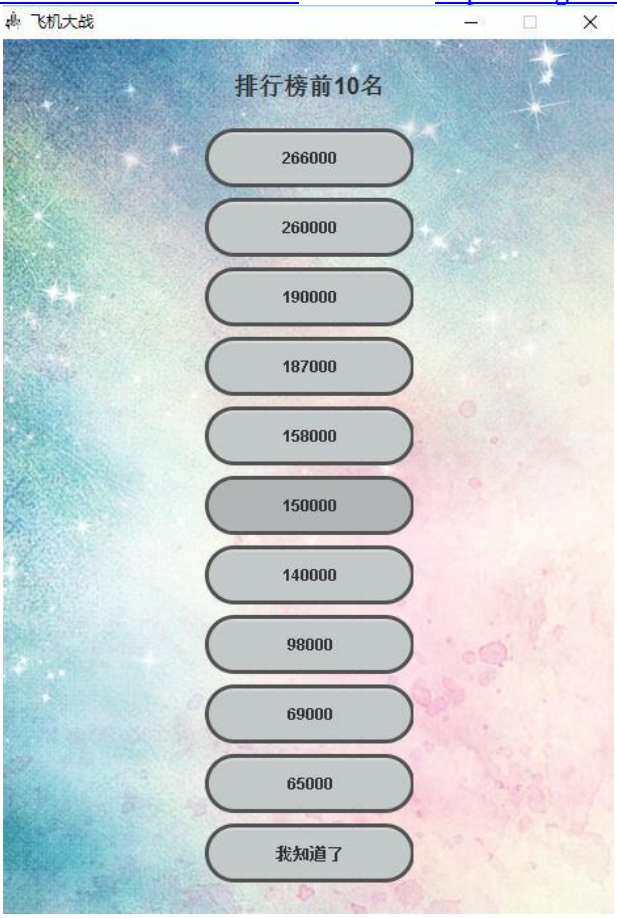
图五：飞机打飞机游戏结束时的界面
（点击是则游戏从新开始，否则返回游戏主界面）

图六：点击炮台打飞机的游戏界面
（左上方第一行为分数，第二行为命数）

图七：获得奖励和奖励作用时间的界面

图八：游戏结束后画面
（点击是则游戏从新开始，否则返回游戏主界面）

图九：点击排行榜后的界面



六、遇到的问题和解决措施

1、子弹飞行问题：一开始设计的是敌军飞机释放子弹，结果发现子弹放射方向与应当方向相反，无法对我军造成伤害，而且太过密集无法躲避。因此，我又定义了 BOMB 类解决这个问题。但是，炮弹仍然发射过多，并且发射的有规律。所以为了解决这个问题，我首先反复调整实验了飞机出现频率和炮弹发射频率的参数，并对炮弹发射的线程函数进行了改进，先发炮弹再进行循环，而不是先循环再发射炮弹，这样，随着敌军随机出现，炮弹也随机出现。

2、碰撞问题：在飞机与敌机或者飞机与炮弹发生碰撞的时候，我发现有时候碰撞了一次却失去了好几条生命，思考过后，我发觉，一开始写的只是图形的碰撞，在飞机的移动过程中，由于图形边缘不规则所以可能出现多次碰撞。为解决这个问题，在第一次碰撞中我就使死去的敌机或炸弹消失，便解决了再次碰撞的问题。

3、截图问题：由于从网上下载的图片居然是连在一起的无法直接引用，所以我设置了 ImageLoader 类实现截出原图的一小部分使用。

4、代码冗余问题：由于敌军飞机种类较多，但是功能相同，所以我建立了 EnemyPlaneFactory 类，让所有敌军飞机的类继承之。

七、需要改进的地方与遗憾

1、没有实习游戏的暂停。小游戏最需要的是及时性，用户可以方便的选择开始，突然有事可以暂停后继续游戏。然而没有实现这个功能，比较遗憾。

2、界面可以再度优化，增加一键换肤功能（点击按钮切换不同的背景）和选择战机功能。

3、两个游戏飞机飞行轨迹单一，可以设计不同方向、不同角度的飞行方式，增加游戏的可玩性。

八、心得体会

本次 java 大作业我们小组选择的题目是炮台打飞机，要求设计有计分有音效有生命值的小游戏。

对于这次实验设计我们开始觉得有点无从下手，因为在课程中学到的大部分是一些概念上的东西，对于整体的软件设计是完全摸不到头绪的。于是我们借阅了很多与 java 实验相关的书籍，看过基础的知识点之后着重研究了小游戏的设计，然后我们对整个有了一个大体思路。诸如此类的问题还有很多，我们在一步一步地修改代码中慢慢地完成了自己的实验，从而进一步掌握了 java 中开发图形应用程序的常用控件、事件处理机制、常用功能如文件读取、数据流的读取和管理、绘图显示的实现。

之前做过 C 语言的相关项目设计，感觉 java 相比 C 最不同的一点就是用户界面的实现要简单很多，实用性更强。此外 java 还是一门比较强大的面向对象的语言，不仅仅因为它的虚拟机功能，可以在不同的平台上运行，而且它取消了 C/C++中的指针、类的多重继承等比较繁琐的内容，让实际编程工作变得更加简单灵活。

通过此次实验操作，我们对软件设计有了一个整体的把握，要考虑到软件的实际需求、算法分析、设计、代码实现和测试，通过设计类图理清各个类成员之间的依赖关系和调用关系，找到实现具体需求的成员方法。在软件代码编

写实现过程中很重要的一点就是请教别人和互相之间的讨论，我们很多问题都是在讨论中解决的，自己单独思考，不仅要花费更多的时间而且最后未必能成功解决，团队之间的合作是非常重要的。再有就是编程这件事亲手实践远比单单只看书本上的知识点要有用的多，只有自己实际操作之后才能发现一些细微的差错和需要注意的事项，从而进一步提高自己的编程水平。

九、小组成员姓名和联系方法

刘美含 手机：15951929378 邮箱：760008522@qq.com
薛睿 手机：18362939179 邮箱：549063417@qq.com
钟铭 手机：17714365160 邮箱：634334680@qq.com

十、对该课程学习的意见和建议

虽然是限选课，但是我们觉得这门课程对我们来说还是很有用很重要的。回顾这个学期，感觉自己在这门课上面花费的时间和精力远远不够，结合自身体会，在这里我们想提的一点建议是：如果把平时成绩中的四个小实验穿插在平时讲课中来做，我们感觉会比全都放在期末做效果好一点。因为在刚学了某块知识之后，做一做实验会巩固一下大家对相关内容的掌握。

附录：软件主要代码

Main.java

```
package com.beancore;
import com.beancore.ui.MainFrame;
public class Main {
    public static void main(String args[]) throws InterruptedException {
       MainFrame mainFrame;
       try {
          mainFrame = new MainFrame();
          mainFrame.loadGame();
       } catch (Exception e) {
          e.printStackTrace();
       }
    }
}
```

BulletType.java

```
package com.beancore.config;

public enum BulletType {
    YELLOW_BULLET, BLUE_BULLET
}
```

CathableWeapon.java

```
package com.beancore.config;

public enum CatchableWeaponType {
    BOMB, DOUBLE_LASER
}
```

BigPlane.java/*飞机打飞机的中型飞机*/

```
package com.beancore.config;

public class ImageConstants {
    public final static int GAME_LOADING_PLANE_1_POS_X = 0;
    public final static int GAME_LOADING_PLANE_1_POS_Y = 38;
    public final static int GAME_LOADING_PLANE_1_WIDTH = 200;
    public final static int GAME_LOADING_PLANE_1_HEIGHT = 37;

    public final static int GAME_LOADING_PLANE_2_POS_X = 0;
    public final static int GAME_LOADING_PLANE_2_POS_Y = 0;
    public final static int GAME_LOADING_PLANE_2_WIDTH = 200;
    public final static int GAME_LOADING_PLANE_2_HEIGHT = 37;

    public final static int GAME_LOADING_PLANE_3_POS_X = 480;
    public final static int GAME_LOADING_PLANE_3_POS_Y = 662;
    public final static int GAME_LOADING_PLANE_3_WIDTH = 200;
    public final static int GAME_LOADING_PLANE_3_HEIGHT = 37;

    public final static int GAME_LOADING_TEXT_IMG_POS_X = 480;
    public final static int GAME_LOADING_TEXT_IMG_POS_Y = 840;
    public final static int GAME_LOADING_TEXT_IMG_WIDTH = 450;
    public final static int GAME_LOADING_TEXT_IMG_HEIGHT = 90;

    public final static int GAME_BACKGROUND_IMG_POS_X = 0;
    public final static int GAME_BACKGROUND_IMG_POS_Y = 75;
    public final static int GAME_BACKGROUND_IMG_WIDTH = 480;
    public final static int GAME_BACKGROUND_IMG_HEIGHT = 852;

    public final static int YELLOW_BULLET_POS_X = 1004;
    public final static int YELLOW_BULLET_POS_Y = 987;
    public final static int YELLOW_BULLET_WIDTH = 9;
    public final static int YELLOW_BULLET_HEIGHT = 21;
```

```
public final static int BLUE_BULLET_POS_X = 71;
public final static int BLUE_BULLET_POS_Y = 78;
public final static int BLUE_BULLET_WIDTH = 7;
public final static int BLUE_BULLET_HEIGHT = 21;

public final static int MY_PLANE_POS_X = 1;
public final static int MY_PLANE_POS_Y = 100;
public final static int MY_PLANE_WIDTH = 100;
public final static int MY_PLANE_HEIGHT = 124;

public final static int MY_PLANE_FLYING_POS_X = 168;
public final static int MY_PLANE_FLYING_POS_Y = 362;
public final static int MY_PLANE_FLYING_WIDTH = 96;
public final static int MY_PLANE_FLYING_HEIGHT = 112;

public final static int SMALL_PLANE_POS_X = 539;
public final static int SMALL_PLANE_POS_Y = 617;
public final static int SMALL_PLANE_WIDTH = 50;
public final static int SMALL_PLANE_HEIGHT = 34;

public final static int BIG_PLANE_POS_X = 1;
public final static int BIG_PLANE_POS_Y = 3;
public final static int BIG_PLANE_WIDTH = 68;
public final static int BIG_PLANE_HEIGHT = 88;

public final static int BOSS_PLANE_POS_X = 336;
public final static int BOSS_PLANE_POS_Y = 751;
public final static int BOSS_PLANE_WIDTH = 168;
public final static int BOSS_PLANE_HEIGHT = 249;

public final static int DOUBLE_LASER_POS_X = 270;
public final static int DOUBLE_LASER_POS_Y = 399;
public final static int DOUBLE_LASER_WIDTH = 55;
public final static int DOUBLE_LASER_HEIGHT = 85;

public final static int BOMB_POS_X = 103;
public final static int BOMB_POS_Y = 119;
public final static int BOMB_WIDTH = 58;
public final static int BOMB_HEIGHT = 106;

public final static int CAUGHT_BOMB_POS_X = 811;
public final static int CAUGHT_BOMB_POS_Y = 694;
public final static int CAUGHT_BOMB_WIDTH = 62;
public final static int CAUGHT_BOMB_HEIGHT = 52;

public final static int SMALL_PLANE_FIGHTING_POS_X = 272;
public final static int SMALL_PLANE_FIGHTING_POS_Y = 356;
public final static int SMALL_PLANE_FIGHTING_WIDTH = 48;
public final static int SMALL_PLANE_FIGHTING_HEIGHT = 35;

public final static int SMALL_PLANE_KILLED_POS_X = 878;
public final static int SMALL_PLANE_KILLED_POS_Y = 704;
public final static int SMALL_PLANE_KILLED_WIDTH = 49;
public final static int SMALL_PLANE_KILLED_HEIGHT = 42;

public final static int SMALL_PLANE_ASHED_POS_X = 935;
public final static int SMALL_PLANE_ASHED_POS_Y = 706;
public final static int SMALL_PLANE_ASHED_WIDTH = 45;
public final static int SMALL_PLANE_ASHED_HEIGHT = 38;

public final static int BIG_PLANE_FIGHTING_POS_X = 432;
public final static int BIG_PLANE_FIGHTING_POS_Y = 529;
public final static int BIG_PLANE_FIGHTING_WIDTH = 69;
```

```
public final static int BIG_PLANE_FIGHTING_HEIGHT = 95;

public final static int BIG_PLANE_HITTED_POS_X = 534;
public final static int BIG_PLANE_HITTED_POS_Y = 657;
public final static int BIG_PLANE_HITTED_WIDTH = 69;
public final static int BIG_PLANE_HITTED_HEIGHT = 95;

public final static int BIG_PLANE_BADDLY_WOUNDED_POS_X = 603;
public final static int BIG_PLANE_BADDLY_WOUNDED_POS_Y = 656;
public final static int BIG_PLANE_BADDLY_WOUNDED_WIDTH = 69;
public final static int BIG_PLANE_BADDLY_WOUNDED_HEIGHT = 95;

public final static int BIG_PLANE_KILLED_POS_X = 672;
public final static int BIG_PLANE_KILLED_POS_Y = 654;
public final static int BIG_PLANE_KILLED_WIDTH = 69;
public final static int BIG_PLANE_KILLED_HEIGHT = 95;

public final static int BIG_PLANE_ASHED_POS_X = 744;
public final static int BIG_PLANE_ASHED_POS_Y = 665;
public final static int BIG_PLANE_ASHED_WIDTH = 64;
public final static int BIG_PLANE_ASHED_HEIGHT = 76;

public final static int BOSS_PLANE_FIGHTING_POS_X = 168;
public final static int BOSS_PLANE_FIGHTING_POS_Y = 753;
public final static int BOSS_PLANE_FIGHTING_WIDTH = 166;
public final static int BOSS_PLANE_FIGHTING_HEIGHT = 256;

public final static int BOSS_PLANE_HITTED_POS_X = 1;
public final static int BOSS_PLANE_HITTED_POS_Y = 489;
public final static int BOSS_PLANE_HITTED_WIDTH = 166;
public final static int BOSS_PLANE_HITTED_HEIGHT = 256;

public final static int BOSS_PLANE_BADDLY_WOUNDED_POS_X = 166;
public final static int BOSS_PLANE_BADDLY_WOUNDED_POS_Y = 487;
public final static int BOSS_PLANE_BADDLY_WOUNDED_WIDTH = 166;
public final static int BOSS_PLANE_BADDLY_WOUNDED_HEIGHT = 256;

public final static int BOSS_PLANE_KILLED_POS_X = 672;
public final static int BOSS_PLANE_KILLED_POS_Y = 751;
public final static int BOSS_PLANE_KILLED_WIDTH = 166;
public final static int BOSS_PLANE_KILLED_HEIGHT = 256;

public final static int BOSS_PLANE_ASHED_POS_X = 1;
public final static int BOSS_PLANE_ASHED_POS_Y = 753;
public final static int BOSS_PLANE_ASHED_WIDTH = 156;
public final static int BOSS_PLANE_ASHED_HEIGHT = 220;

public final static int SCORE_IMG_POS_X = 271;
public final static int SCORE_IMG_POS_Y = 248;
public final static int SCORE_IMG_WIDTH = 50;
public final static int SCORE_IMG_HEIGHT = 50;

public final static int NUMBER_0_POS_X = 179;
public final static int NUMBER_0_POS_Y = 0;
public final static int NUMBER_0_WIDTH = 26;
public final static int NUMBER_0_HEIGHT = 38;

public final static int NUMBER_1_POS_X = 272;
public final static int NUMBER_1_POS_Y = 0;
public final static int NUMBER_1_WIDTH = 6;
public final static int NUMBER_1_HEIGHT = 38;

public final static int NUMBER_2_POS_X = 150;
```

```
    public final static int NUMBER_2_POS_Y = 0;
    public final static int NUMBER_2_WIDTH = 28;
    public final static int NUMBER_2_HEIGHT = 38;

    public final static int NUMBER_3_POS_X = 30;
    public final static int NUMBER_3_POS_Y = 0;
    public final static int NUMBER_3_WIDTH = 25;
    public final static int NUMBER_3_HEIGHT = 38;

    public final static int NUMBER_4_POS_X = 0;
    public final static int NUMBER_4_POS_Y = 0;
    public final static int NUMBER_4_WIDTH = 28;
    public final static int NUMBER_4_HEIGHT = 38;

    public final static int NUMBER_5_POS_X = 236;
    public final static int NUMBER_5_POS_Y = 0;
    public final static int NUMBER_5_WIDTH = 34;
    public final static int NUMBER_5_HEIGHT = 38;

    public final static int NUMBER_6_POS_X = 206;
    public final static int NUMBER_6_POS_Y = 0;
    public final static int NUMBER_6_WIDTH = 30;
    public final static int NUMBER_6_HEIGHT = 38;

    public final static int NUMBER_7_POS_X = 278;
    public final static int NUMBER_7_POS_Y = 0;
    public final static int NUMBER_7_WIDTH = 32;
    public final static int NUMBER_7_HEIGHT = 38;

    public final static int NUMBER_8_POS_X = 87;
    public final static int NUMBER_8_POS_Y = 0;
    public final static int NUMBER_8_WIDTH = 30;
    public final static int NUMBER_8_HEIGHT = 38;

    public final static int NUMBER_9_POS_X = 120;
    public final static int NUMBER_9_POS_Y = 0;
    public final static int NUMBER_9_WIDTH = 28;
    public final static int NUMBER_9_HEIGHT = 38;

    public final static int X_MARK_POS_X = 58;
    public final static int X_MARK_POS_Y = 0;
    public final static int X_MARK_WIDTH = 28;
    public final static int X_MARK_HEIGHT = 38;

}
```

BigPlane2.java/*炮台打飞机中的中型飞机*/

```
package com.beancore.entity;

import java.awt.Graphics;
import java.awt.Image;
import java.util.List;

import com.beancore.config.BulletType;
import com.beancore.config.Config;
import com.beancore.config.EnemyPlaneType;
import com.beancore.config.ImageConstants;
import com.beancore.ui.GamePlayingPanel2;
import com.beancore.util.Images;

public class BigPlane2 extends EnemyPlane2 {

    public BigPlane2(GamePlayingPanel2 getPlayingPanel, EnemyPlaneType enemyType)
{
```

```
   super(getPlayingPanel, enemyType);
}
 public void drawFighting(Graphics g) {
   new Thread(new DrawFighting(g)).start();
}


public void drawKilled(Graphics g) {
   new Thread(new DrawKilled(g)).start();
}

class DrawFighting implements Runnable {
   private Graphics g;

   DrawFighting(Graphics g) {
      this.g = g;
   }

   public void run() {
      drawFightingRun(g);
   }
}

class DrawKilled implements Runnable {
   private Graphics g;

   DrawKilled(Graphics g) {
      this.g = g;
   }

   public void run() {
      drawKilledRun(g);
   }

}

public void drawFightingRun(Graphics g) {
   this.setPlaneImage(Images.BIG_PLANE_FIGHTING_IMG);
   this.setWidth(ImageConstants.BIG_PLANE_FIGHTING_WIDTH);
   this.setHeight(ImageConstants.BIG_PLANE_FIGHTING_HEIGHT);
   super.draw(g);
   try {
      Thread.sleep(Config.BIG_PLANE_STATUS_CHANGE_INTERVAL);
   } catch (InterruptedException e) {

   }
}

public void drawKilledRun(Graphics g) {
   this.setPlaneImage(Images.BIG_PLANE_HITTED_IMG);
   this.setWidth(ImageConstants.BIG_PLANE_HITTED_WIDTH);
   this.setHeight(ImageConstants.BIG_PLANE_HITTED_HEIGHT);
   super.draw(g);
   try {
      Thread.sleep(Config.BIG_PLANE_STATUS_CHANGE_INTERVAL);
   } catch (InterruptedException e) {

   }

   this.setPlaneImage(Images.BIG_PLANE_BADDLY_WOUNDED_IMG);
   this.setWidth(ImageConstants.BIG_PLANE_BADDLY_WOUNDED_WIDTH);
   this.setHeight(ImageConstants.BIG_PLANE_BADDLY_WOUNDED_HEIGHT);
   super.draw(g);
```

```
        try {
            Thread.sleep(Config.BIG_PLANE_STATUS_CHANGE_INTERVAL);
        } catch (InterruptedException e) {

        }

        this.setPlaneImage(Images.BIG_PLANE_KILLED_IMG);
        this.setWidth(ImageConstants.BIG_PLANE_KILLED_WIDTH);
        this.setHeight(ImageConstants.BIG_PLANE_KILLED_HEIGHT);
        super.draw(g);
        try {
            Thread.sleep(Config.BIG_PLANE_STATUS_CHANGE_INTERVAL);
        } catch (InterruptedException e) {

        }

        this.setPlaneImage(Images.BIG_PLANE_ASHED_IMG);
        this.setWidth(ImageConstants.BIG_PLANE_ASHED_WIDTH);
        this.setHeight(ImageConstants.BIG_PLANE_ASHED_HEIGHT);
        super.draw(g);
        try {
            Thread.sleep(Config.BIG_PLANE_STATUS_CHANGE_INTERVAL);
        } catch (InterruptedException e) {

        }
    }

}
Bomb2.java/*炮台打飞机中的炸弹*/
package com.beancore.entity;

import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Image;
import java.awt.Rectangle;
import java.util.List;

import com.beancore.config.Config;
import com.beancore.config.ImageConstants;
import com.beancore.listener.BombListener;
import com.beancore.ui.GamePlayingPanel2;
import com.beancore.util.Images;

public class Bomb2{
      private int posX;
    private int posY;
    private int width;
    private int height;
    private int speed;

    private GamePlayingPanel2 gamePlayingPanel;
    private BombListener listener;
    private Image bombImage;

    public Bomb2(GamePlayingPanel2 gamePlayingPanel) {
        this.gamePlayingPanel = gamePlayingPanel;
          bombImage = Images.BOMB_IMG;
          width = ImageConstants.BOMB_WIDTH;
          height = ImageConstants.BOMB_HEIGHT;
          speed = 40;
    }

    public Rectangle getRectangle() {
```

```
    return new Rectangle(posX, posY, width, height);
}

public void draw(Graphics g) {
    Graphics2D g2d = (Graphics2D) g;
    g2d.drawImage(bombImage, posX, posY, width, height, gamePlayingPanel);

}

public void addBombListener(GamePlayingPanel2 playingPanel) {
    this.gamePlayingPanel = playingPanel;
}

public int getPosX() {
    return posX;
}

public void setPosX(int posX) {
    this.posX = posX;
}

public int getPosY() {
    return posY;
}

public void setPosY(int posY) {
    this.posY = posY;
}

public int getWidth() {
    return width;
}

public void setWidth(int width) {
    this.width = width;
}

public int getHeight() {
    return height;
}

public void setHeight(int height) {
    this.height = height;
}


public GamePlayingPanel2 getGamePlayingPanel() {
    return gamePlayingPanel;
}

public void setGamePlayingPanel(GamePlayingPanel2 gamePlayingPanel) {
    this.gamePlayingPanel = gamePlayingPanel;
}

public BombListener getListener() {
    return listener;
}

public void setListener(BombListener listener) {
    this.listener = listener;
}

public Image getBulletImage() {
```

```
        return bombImage;
    }

    public void setBombImage(Image bombImage) {
        this.bombImage = bombImage;
    }

    public int getSpeed() {
        return speed;
    }

    public void setSpeed(int speed) {
        this.speed = speed;
    }

    public MyPlane2 hitMyPlanes() {
        MyPlane2 myPlane = this.gamePlayingPanel.getMyPlane();
            if (this.getRectangle().intersects(myPlane.getRectange())) {
                myPlane.addHittedCount();
                return myPlane;
        }
        return null;
         }
}
Bullet2.java/*炮台打飞机中的子弹*/
package com.beancore.entity;

import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Image;
import java.awt.Rectangle;
import java.util.List;

import com.beancore.config.BulletType;
import com.beancore.config.Config;
import com.beancore.config.ImageConstants;
import com.beancore.listener.BulletListener;
import com.beancore.ui.GamePlayingPanel;
import com.beancore.util.Images;

public class Bullet {
    private int posX;
    private int posY;
    private int width;
    private int height;
    private int speed;
    private BulletType bulletType;

    private GamePlayingPanel gamePlayingPanel;
    private BulletListener listener;
    private Image bulletImage;

    public Bullet(GamePlayingPanel gamePlayingPanel, BulletType bulletType) {
        this.gamePlayingPanel = gamePlayingPanel;
        this.bulletType = bulletType;
        switch (this.bulletType) {
        case YELLOW_BULLET:
            bulletImage = Images.YELLOW_BULLET_IMG;
            width = ImageConstants.YELLOW_BULLET_WIDTH;
            height = ImageConstants.YELLOW_BULLET_HEIGHT;
            speed = Config.YELLOW_BULLET_MOVE_SPEED;
            break;
        case BLUE_BULLET:
```

```
            bulletImage = Images.BLUE_BULLET_IMG;
            width = ImageConstants.YELLOW_BULLET_WIDTH;
            height = ImageConstants.YELLOW_BULLET_HEIGHT;
            speed = Config.BLUE_BULLET_MOVE_SPEED;
            break;
        }
    }

    public Rectangle getRectangle() {
        return new Rectangle(posX, posY, width, height);
    }

    public void draw(Graphics g) {
        Graphics2D g2d = (Graphics2D) g;
        g2d.drawImage(bulletImage, posX, posY, width, height, gamePlayingPanel);

    }

    public EnemyPlane hitEnemyPlanes() {
        List<EnemyPlane> enmeyPlanes = this.gamePlayingPanel.getEnemyPlanes();
        for (int i = 0; i < enmeyPlanes.size(); i++) {
            EnemyPlane enemyPlane = enmeyPlanes.get(i);
            if (this.getRectangle().intersects(enemyPlane.getRectangle())) {
                enemyPlane.addHittedCount();
                return enemyPlane;
            }
        }
        return null;
    }

    public void addBulletListener(GamePlayingPanel playingPanel) {
        this.gamePlayingPanel = playingPanel;
    }

    public int getPosX() {
        return posX;
    }

    public void setPosX(int posX) {
        this.posX = posX;
    }

    public int getPosY() {
        return posY;
    }

    public void setPosY(int posY) {
        this.posY = posY;
    }

    public int getWidth() {
        return width;
    }

    public void setWidth(int width) {
        this.width = width;
    }

    public int getHeight() {
        return height;
    }

    public void setHeight(int height) {
```

```
        this.height = height;
    }

    public BulletType getBulletType() {
        return bulletType;
    }

    public void setBulletType(BulletType bulletType) {
        this.bulletType = bulletType;
    }

    public GamePlayingPanel getGamePlayingPanel() {
        return gamePlayingPanel;
    }

    public void setGamePlayingPanel(GamePlayingPanel gamePlayingPanel) {
        this.gamePlayingPanel = gamePlayingPanel;
    }

    public BulletListener getListener() {
        return listener;
    }

    public void setListener(BulletListener listener) {
        this.listener = listener;
    }

    public Image getBulletImage() {
        return bulletImage;
    }

    public void setBulletImage(Image bulletImage) {
        this.bulletImage = bulletImage;
    }

    public int getSpeed() {
        return speed;
    }

    public void setSpeed(int speed) {
        this.speed = speed;
    }

}
```

CathableWeapon2.java/*炮弹打飞机的奖励*/

```
package com.beancore.entity;

import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Image;
import java.awt.Rectangle;

import com.beancore.config.Config;
import com.beancore.config.EnemyPlaneType;
import com.beancore.listener.EnemyPlaneListener2;
import com.beancore.ui.GamePlayingPanel2;
import com.beancore.factory. BombFactory;

public abstract class EnemyPlane2{
    private int posX;
    private int posY;
    private int width;
    private int height;
```

```
    private int speed;
    private int hittedCount;
    private int killedCount;
    private int killedScore;
    private Image planeImage;
    private EnemyPlaneListener2 listener;
    private EnemyPlaneType enemyType;
    private GamePlayingPanel2 gamePlayingPanel;

    public EnemyPlane2(GamePlayingPanel2 getPlayingPanel, EnemyPlaneType
enemyType) {
       this.gamePlayingPanel = getPlayingPanel;
       this.enemyType = enemyType;
       this.hittedCount = 0;
       new Thread(new LauchBombThread()).start();
    }

    public Rectangle getRectangle() {//获得坐标
       return new Rectangle(posX, posY, width, height);
    }

    public void draw(Graphics g) {
       Graphics2D g2d = (Graphics2D) g;
       g2d.drawImage(planeImage, posX, posY, width, height, gamePlayingPanel);
    }

    public void addEnemyPlaneListener(EnemyPlaneListener2 listener) {
       this.listener = listener;
    }

    public void addHittedCount() {
       this.hittedCount++;
    }

    public boolean isKilled() {
       return this.hittedCount >= this.killedCount;
    }

    public abstract void drawFighting(Graphics g);

    public abstract void drawKilled(Graphics g);

    public EnemyPlaneType getEnemyType() {
       return enemyType;
    }

    public void setEnemyType(EnemyPlaneType enemyType) {
       this.enemyType = enemyType;
    }

    public GamePlayingPanel2 getGamePlayingPanel() {
       return gamePlayingPanel;
    }

    public void setGamePlayingPanel(GamePlayingPanel2 gamePlayingPanel) {
       this.gamePlayingPanel = gamePlayingPanel;
    }

    public int getPosX() {
       return posX;
    }

    public void setPosX(int posX) {
```

```
    this.posX = posX;
}

public int getPosY() {
    return posY;
}

public void setPosY(int posY) {
    this.posY = posY;
}

public int getWidth() {
    return width;
}

public void setWidth(int width) {
    this.width = width;
}

public int getHeight() {
    return height;
}

public void setHeight(int height) {
    this.height = height;
}

public int getSpeed() {
    return speed;
}

public void setSpeed(int speed) {
    this.speed = speed;
}

public EnemyPlaneListener2 getListener() {
    return listener;
}

public void setListener(EnemyPlaneListener2 listener) {
    this.listener = listener;
}

public Image getPlaneImage() {
    return planeImage;
}

public void setPlaneImage(Image planeImage) {
    this.planeImage = planeImage;
}

public int getHittedCount() {
    return hittedCount;
}

public void setHittedCount(int hittedCount) {
    this.hittedCount = hittedCount;
}

public int getKilledCount() {
    return killedCount;
}
```

```
    public void setKilledCount(int killedCount) {
       this.killedCount = killedCount;
    }

    public int getKilledScore() {
       return killedScore;
    }

    public void setKilledScore(int killedScore) {
       this.killedScore = killedScore;
    }

    public void lauchBomb() {
       if (true) {
             switch (this.getEnemyType()) {
          case SMALL_ENEMY_PLANE:
             break;
          case BIG_ENEMY_PLANE:
             Bomb2 bomb = BombFactory.createBomb(this);
                   bomb.addBombListener(this.gamePlayingPanel);
                   synchronized (this.gamePlayingPanel.getBombs()) {
                   this.gamePlayingPanel.getBombs().add(bomb);
                }
             break;
          case BOSS_ENEMY_PLANE:
             Bomb2 bomb2 = BombFactory.createBomb(this);
                   bomb2.addBombListener(this.gamePlayingPanel);
                   synchronized (this.gamePlayingPanel.getBombs()) {
                      this.gamePlayingPanel.getBombs().add(bomb2);
                }
             break;
          }

       }
}

       class LauchBombThread implements Runnable {
       public void run() {
          while(true){
             lauchBomb();
             try {
                Thread.sleep(Config.BULLET_FIRE_INTERVAL+10000);
             } catch (InterruptedException e) {

             }

          }
       }
       }

}
EnemyPlane2.java/*炮弹打飞机的敌机*/
package com.beancore.entity;

import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Image;
import java.awt.Rectangle;

import com.beancore.config.Config;
import com.beancore.config.EnemyPlaneType;
import com.beancore.listener.EnemyPlaneListener2;
import com.beancore.ui.GamePlayingPanel2;
```

```
import com.beancore.factory. BombFactory;

public abstract class EnemyPlane2{
    private int posX;
    private int posY;
    private int width;
    private int height;
    private int speed;
    private int hittedCount;
    private int killedCount;
    private int killedScore;
    private Image planeImage;
    private EnemyPlaneListener2 listener;
    private EnemyPlaneType enemyType;
    private GamePlayingPanel2 gamePlayingPanel;

    public EnemyPlane2(GamePlayingPanel2 getPlayingPanel, EnemyPlaneType
enemyType) {
       this.gamePlayingPanel = getPlayingPanel;
       this.enemyType = enemyType;
       this.hittedCount = 0;
       new Thread(new LauchBombThread()).start();
    }

    public Rectangle getRectangle() {//获得坐标
       return new Rectangle(posX, posY, width, height);
    }

    public void draw(Graphics g) {
       Graphics2D g2d = (Graphics2D) g;
       g2d.drawImage(planeImage, posX, posY, width, height, gamePlayingPanel);
    }

    public void addEnemyPlaneListener(EnemyPlaneListener2 listener) {
       this.listener = listener;
    }

    public void addHittedCount() {
       this.hittedCount++;
    }

    public boolean isKilled() {
       return this.hittedCount >= this.killedCount;
    }

    public abstract void drawFighting(Graphics g);

    public abstract void drawKilled(Graphics g);

    public EnemyPlaneType getEnemyType() {
       return enemyType;
    }

    public void setEnemyType(EnemyPlaneType enemyType) {
       this.enemyType = enemyType;
    }

    public GamePlayingPanel2 getGamePlayingPanel() {
       return gamePlayingPanel;
    }

    public void setGamePlayingPanel(GamePlayingPanel2 gamePlayingPanel) {
       this.gamePlayingPanel = gamePlayingPanel;
```

```
}

public int getPosX() {
   return posX;
}

public void setPosX(int posX) {
   this.posX = posX;
}

public int getPosY() {
   return posY;
}

public void setPosY(int posY) {
   this.posY = posY;
}

public int getWidth() {
   return width;
}

public void setWidth(int width) {
   this.width = width;
}

public int getHeight() {
   return height;
}

public void setHeight(int height) {
   this.height = height;
}

public int getSpeed() {
   return speed;
}

public void setSpeed(int speed) {
   this.speed = speed;
}

public EnemyPlaneListener2 getListener() {
   return listener;
}

public void setListener(EnemyPlaneListener2 listener) {
   this.listener = listener;
}

public Image getPlaneImage() {
   return planeImage;
}

public void setPlaneImage(Image planeImage) {
   this.planeImage = planeImage;
}

public int getHittedCount() {
   return hittedCount;
}

public void setHittedCount(int hittedCount) {
```

```
        this.hittedCount = hittedCount;
    }

    public int getKilledCount() {
        return killedCount;
    }

    public void setKilledCount(int killedCount) {
        this.killedCount = killedCount;
    }

    public int getKilledScore() {
        return killedScore;
    }

    public void setKilledScore(int killedScore) {
        this.killedScore = killedScore;
    }

    public void lauchBomb() {
        if (true) {
                switch (this.getEnemyType()) {
            case SMALL_ENEMY_PLANE:
                break;
            case BIG_ENEMY_PLANE:
                Bomb2 bomb = BombFactory.createBomb(this);
                      bomb.addBombListener(this.gamePlayingPanel);
                      synchronized (this.gamePlayingPanel.getBombs()) {
                      this.gamePlayingPanel.getBombs().add(bomb);
                    }
                break;
            case BOSS_ENEMY_PLANE:
                Bomb2 bomb2 = BombFactory.createBomb(this);
                      bomb2.addBombListener(this.gamePlayingPanel);
                      synchronized (this.gamePlayingPanel.getBombs()) {
                          this.gamePlayingPanel.getBombs().add(bomb2);
                    }
                break;
            }

        }
}

        class LauchBombThread implements Runnable {
        public void run() {
            while(true){
                lauchBomb();
                try {
                    Thread.sleep(Config.BULLET_FIRE_INTERVAL+10000);
                } catch (InterruptedException e) {

                }

            }
        }
        }

}
MyPlane.java/*飞机打飞机的我方飞机*/
package com.beancore.entity;

import java.awt.Graphics;
import java.awt.Graphics2D;
```

```
import java.awt.Image;
import java.awt.Rectangle;
import java.awt.event.MouseEvent;
import java.util.LinkedList;
import java.util.List;

import com.beancore.config.BulletType;
import com.beancore.config.Config;
import com.beancore.config.ImageConstants;
import com.beancore.factory.BulletFactory;
import com.beancore.ui.GamePlayingPanel;
import com.beancore.util.Images;

public class MyPlane {

      //当前坐标
    private int posX;
    private int posY;

    private int width;
    private int height;
    private Image planeImage;
    private Image planeFlyingImage;
    private boolean isAlive;
    private boolean hitDoubleLaser;
    private List<Bomb> holdBombList;
    private BulletType bulletType;
    private GamePlayingPanel playingPanel;
    private boolean flip;

    public MyPlane(GamePlayingPanel gamePlayingPanel) {
       this.isAlive = true;
       this.flip = true;
       this.playingPanel = gamePlayingPanel;
       this.width = ImageConstants.MY_PLANE_WIDTH;
       this.height = ImageConstants.MY_PLANE_HEIGHT;
       this.planeImage = Images.MY_PLANE_IMG;
       this.planeFlyingImage = Images.MY_PLANE_FLYING_IMG;
       this.holdBombList = new LinkedList<Bomb>();
       new Thread(new LauchBulletThread()).start();
    }

    public Rectangle getRectange() {
       int fix = width / 3;
       return new Rectangle(posX + fix, posY, width / 3, height);
    }

    public void draw(Graphics g) {
       Graphics2D g2d = (Graphics2D) g;
       if (flip) {
          g2d.drawImage(planeImage, posX, posY, width, height, playingPanel);
       } else {
          g2d.drawImage(planeFlyingImage, posX, posY, width, height,
playingPanel);
       }
       flip = !flip;
    }

    public void mouseMoved(MouseEvent e) {
       int x = e.getX();
       int y=e.getY();
       posX = x - width / 2;
       posY = y-height / 2;//垂直位置不动
```

```
    }

    public void lauchBullet() {
       if (isAlive) {
          if (hitDoubleLaser) {
             Bullet bullets[] = BulletFactory.createBlueBullet(this);
             for (Bullet bullet : bullets) {
                bullet.addBulletListener(this.playingPanel);
                synchronized (this.playingPanel.getBullets()) {
                   this.playingPanel.getBullets().add(bullet);
                }
             }
          } else {
             Bullet bullet = BulletFactory.createYellowBullet(this);
             bullet.addBulletListener(this.playingPanel);
             synchronized (this.playingPanel.getBullets()) {
                this.playingPanel.getBullets().add(bullet);
             }
          }
       }
    }

class LauchBulletThread implements Runnable {
       public void run() {
          while (isAlive) {
             try {
                Thread.sleep(Config.BULLET_FIRE_INTERVAL);
             } catch (InterruptedException e) {

             }
             lauchBullet();
          }
       }
    }

    public int getHoldBombCount() {
       return this.holdBombList.size();
    }

    public int getPosX() {
       return posX;
    }

    public void setPosX(int posX) {
       this.posX = posX;
    }

    public int getPosY() {
       return posY;
    }

    public void setPosY(int posY) {
       this.posY = posY;
    }

    public int getWidth() {
       return width;
    }

    public void setWidth(int width) {
       this.width = width;
    }
```

```
    public int getHeight() {
       return height;
    }

    public void setHeight(int height) {
       this.height = height;
    }

    public Image getPlaneImage() {
       return planeImage;
    }

    public void setPlaneImage(Image planeImage) {
       this.planeImage = planeImage;
    }

    public boolean isAlive() {
       return isAlive;
    }

    public void setAlive(boolean isAlive) {
       this.isAlive = isAlive;
    }

    public boolean isHitDoubleLaser() {
       return hitDoubleLaser;
    }

    public void setHitDoubleLaser(boolean hitDoubleLaser) {
       this.hitDoubleLaser = hitDoubleLaser;
    }

    public List<Bomb> getHoldBombList() {
       return holdBombList;
    }

    public void setHoldBombList(List<Bomb> holdBombList) {
       this.holdBombList = holdBombList;
    }
    public BulletType getBulletType() {
       return bulletType;
    }

    public void setBulletType(BulletType bulletType) {
       this.bulletType = bulletType;
    }

    public GamePlayingPanel getPlayingPanel() {
       return playingPanel;
    }
    public void setPlayingPanel(GamePlayingPanel playingPanel) {
       this.playingPanel = playingPanel;
    }
}
EnemyPlaneFactory.java
package com.beancore.factory;

import java.util.Random;

import com.beancore.config.Config;
import com.beancore.config.EnemyPlaneType;
import com.beancore.config.ImageConstants;
import com.beancore.entity.BigPlane;
```

```
import com.beancore.entity.BossPlane;
import com.beancore.entity.EnemyPlane;
import com.beancore.entity.SmallPlane;
import com.beancore.ui.GamePlayingPanel;
import com.beancore.util.Images;

public class EnemyPlaneFactory {
    public static final Random rand = new Random();

      public static final EnemyPlane createEnemyPlane(GamePlayingPanel
playingPanel, EnemyPlaneType enemyPlaneType) {
      EnemyPlane enemyPlane = null;
      switch (enemyPlaneType) {
      case SMALL_ENEMY_PLANE:
          enemyPlane = new SmallPlane(playingPanel, enemyPlaneType);
          enemyPlane.setWidth(ImageConstants.SMALL_PLANE_WIDTH);
          enemyPlane.setHeight(ImageConstants.SMALL_PLANE_HEIGHT);
          enemyPlane.setPlaneImage(Images.SMALL_PLANE_IMG);
          enemyPlane.setKilledCount(Config.BULLET_COUNT_TO_KILL_SMALL_PLANE);
          enemyPlane.setKilledScore(Config.KILL_SMALL_PLANE_SCORE);
          break;
      case BIG_ENEMY_PLANE:
          enemyPlane = new BigPlane(playingPanel, enemyPlaneType);
          enemyPlane.setWidth(ImageConstants.BIG_PLANE_WIDTH);
          enemyPlane.setHeight(ImageConstants.BIG_PLANE_HEIGHT);
          enemyPlane.setPlaneImage(Images.BIG_PLANE_IMG);
          enemyPlane.setKilledCount(Config.BULLET_COUNT_TO_KILL_BIG_PLANE);
          enemyPlane.setKilledScore(Config.KILL_BIG_PLANE_SCORE);
          break;
      case BOSS_ENEMY_PLANE:
          enemyPlane = new BossPlane(playingPanel, enemyPlaneType);
          enemyPlane.setWidth(ImageConstants.BOSS_PLANE_WIDTH);
          enemyPlane.setHeight(ImageConstants.BOSS_PLANE_HEIGHT);
          enemyPlane.setPlaneImage(Images.BOSS_PLANE_IMG);
          enemyPlane.setKilledCount(Config.BULLET_COUNT_TO_KILL_BOSS_PLANE);
          enemyPlane.setKilledScore(Config.KILL_BOSS_PLANE_SCORE);
          break;
      }

      int posX = rand.nextInt(playingPanel.getWidth() - enemyPlane.getWidth());
      int posY = 0;
      enemyPlane.setPosX(posX);
      enemyPlane.setPosY(posY);
      int speed = rand.nextInt(Config.ENEMY_PLANE_MOVE_SPEED_MAX -
Config.ENEMY_PLANE_MOVE_SPEED_MIN)
            + Config.ENEMY_PLANE_MOVE_SPEED_MIN;
      enemyPlane.setSpeed(speed);
      enemyPlane.addEnemyPlaneListener(playingPanel);
      return enemyPlane;
    }
}
```