# 飞机大战项目逻辑分析

### Shoot射击游戏第一天:

1.创建6个对象类，并创建World类测试

### **Shoot射击游戏第二天:**

1.创建了6个对象类的构造方法，在World类中测试

### **Shoot射击游戏第三天:**

1.设计小敌机数组、大敌机数组、小蜜蜂数组、子弹数组，并测试

2.设计FlyingObject超类，6个对象类继承超类

3.给FlyingObject超类设计了两个构造方法，6个派生类中分别调用

### **Shoot射击游戏第四天:**

1.将小敌机数组、大敌机数组、小蜜蜂数组合为FlyingObject数组

2.在6个派生类中重写step()

3.画窗口

### **Shoot射击游戏第五天:**

1.将类中的成员添加访问控制修饰符

2.设置图片属性

### **Shoot射击游戏第六天:**

1.画对象:

1)想画对象，得先去获取对象的图片，

每个对象都得获取图片，意味着获取图片行为为共有的行为，

所以需要将获取图片的行为设计在FlyingObject超类中，

每个对象获取图片的行为都是不一样的，

所以获取图片的行为设计为抽象方法

--------在FlyingObject类中设计抽象方法getImage()

2)在获取对象的图片时，需要去考虑对象的状态(生命周期)，

状态分三种(活着的、死了的(未删除)、删除的)，

在实际应用中，状态都应设计为常量，

每个对象都有状态，所以将状态设计在FlyingObject超类中，

--------在FlyingObject类中设计三个常量和变量(存储当前状态)

在获取图片时，需要去判断每个对象的状态，用于获取不同的图片，

所以对象状态判断的行为为共有的行为，设计在FlyingObject超类中，

每个对象判断状态的行为都是一样的，设计为普通方法

--------在FlyingObject类中设计isLife()/isDead()/isRemove()

3)在派生类中重写getImage()用于在不同状态下获取不同的图片

3.1)Sky天空，直接返回image

3.2)Bullet子弹:

3.2.1)若活着的，直接返回image

3.2.2)若死了的，就删除

3.3)Airplane小敌机:

3.3.1)若活着呢，返回第1张图片

3.3.2)若死了呢，从第2张图到第5张图轮换，第5张图后可以删除了

3.4)Airplane小敌机:

3.4.1)若活着呢，返回第1张图片

3.4.2)若死了呢，从第2张图到第5张图轮换，第5张图后可以删除了

3.5)Airplane小敌机:

3.5.1)若活着呢，返回第1张图片

3.5.2)若死了呢，从第2张图到第5张图轮换，第5张图后可以删除了

3.6)Hero英雄机:

3.6.1)若活着呢，返回2张图片的切换

4)图片有了，就可以开画了，6个对象都可开画，属于共有的行为，

所以将画的行为设计在FlyingObject超类中，

所有对象画的行为都是一样的，所以设计为普通方法

--------在FlyingObject超类中设计普通方法paintObject()画对象

5)因为Sky天空是两张图，所以需要重写超类的paintObject()

--------在Sky类中重写paintObject()

6)画的行为完以后，只需要在窗口上调用即可

--------在World类中重写paint()，而后调用paintObject()

### **Shoot射击游戏第七天:**

1.敌人入场:

1)敌人是由窗口产生的，所以在World中设计nextOne()生成敌人对象

2)敌人入场为定时发生的，在run()中调用enterAction()实现敌人入场

在enterAction()中:

每400毫秒调用nextOne()获取敌人，enemies扩容，装在最后一个元素上

2.子弹入场:

1)子弹是由英雄机发射出来的，所以在Hero中设计shoot()生成子弹对象

2)子弹入场为定时发生的，在run()中调用shootAction()实现子弹入场

在shootAction()中:

每300毫秒调用hero的shoot()获取子弹，bullets扩容，数组追加

3.飞行物移动:

1)移动是所有对象所共有的行为，并且每个对象移动的形为都是不一样的，

所以在超类FlyingObject中设计了抽象方法step()，派生类重写step()

2)飞行物移动为定时发生的，在run()中调用stepAction()实现飞行物移动

在stepAction()中:

天空动，遍历敌人而后敌人动，遍历子弹而后子弹动

### **Shoot射击游戏第八天:**

1.英雄机随着鼠标移动:

1)英雄机移动是英雄机的行为，所以在Hero中设计moveTo(x,y)实现英雄机随着鼠标移动

2)英雄机随着鼠标移动是事件触发的，在侦听器中重写mouseMoved()鼠标移动事件

在mouseMoved()中:

获取鼠标的x和y坐标，而后英雄机随着动

2.删除越界的飞行物:

1)越界检查为所有飞行物都具备的行为，只是每个对象的具体行为不一样，

在超类FlyingObject中设计抽象方法outOfBounds()，6个派生类中分别重写

2)删除越界飞行物为定时发生的，所以在run()中调用outOfBoundsAction()

在outOfBoundsAction()中:

声明不越界敌人/子弹数组，遍历enemies/bullets，获取每个敌人或子弹

判断若不越界，则将对象添加到不越界敌人/子弹数组中，

将不越界敌人/子弹数组复制到enemies/bullets中

3.设计Enemy得分接口，而后Airplane和BigAirplane实现接口

设计Award奖励接口，而后Bee实现接口

### **Shoot射击游戏第九天:**

1.英雄机与敌人的碰撞:

1)借用FlyingObject中的hit()碰撞行为

借用FlyingObject中的goDead()去死行为

在Hero中设计subtractLife()减命、clearDoubleFire()清空火力值

2)因为英雄机与敌人碰撞为定时发生的，所以在run()中调用heroBangAction()实现碰撞

在heroBangAction()中:

遍历所有敌人，获取每个敌人，判断是否与英雄机撞上了，

若撞上了: 敌人去死、英雄机减命、英雄机清空火力值

2.检测游戏结束:

1)借用Hero的getLife()获取命

2)因为检测游戏结束为定时发生的，所以在run()中调用checkGameOver()实现检测结束

在checkGameOverAction()中:

判断英雄机的命数是否<=0，若为true则...

3.画状态:

1)设计了四个状态常量，当前状态state变量

设计了三个静态图片，在static块中初始化

2)在paint()中设计在不同状态下画不同的图片

3)将一堆action()设计为仅在运行状态下执行

将英雄机随着鼠标动设计为仅在运行状态下执行

4)重写mouseClicked()鼠标点击事件

重写mouseExited()鼠标点击事件

重写mouseEntered()鼠标点击事件

### **Shoot射击游戏第十天:**

1)英雄机与敌人的碰撞

2)检测游戏结束

3)画状态