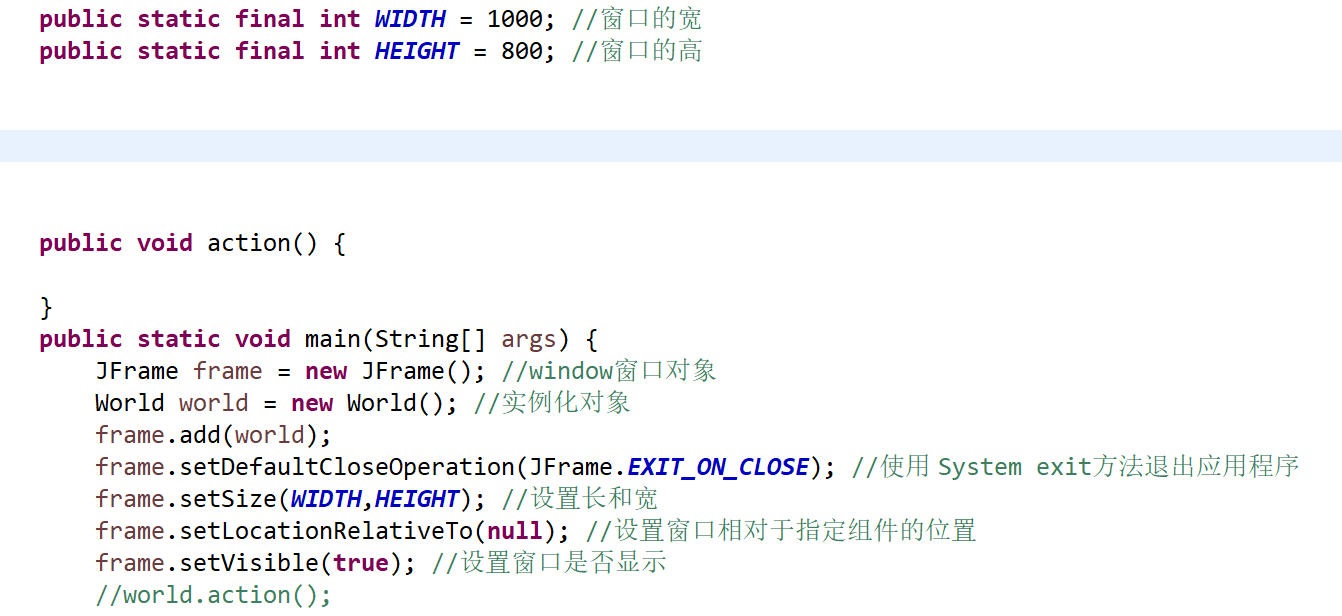
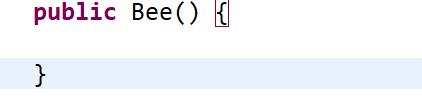
飞机大战

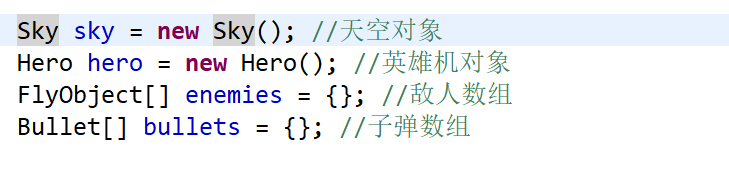
1. 创建了6个对象类，创建World类并测试
2. 找对象:英雄机、大敌机、小敌机、小蜜蜂、子弹、天空
3. 抽类:Hero、BigAirplane、Airplane、Bee、Bullet、Sky
4. 世界类：World



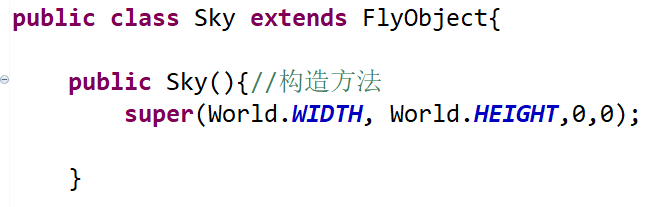
1. 添加构造方法，并创建数组
2. 给6个对象类添加构造方法



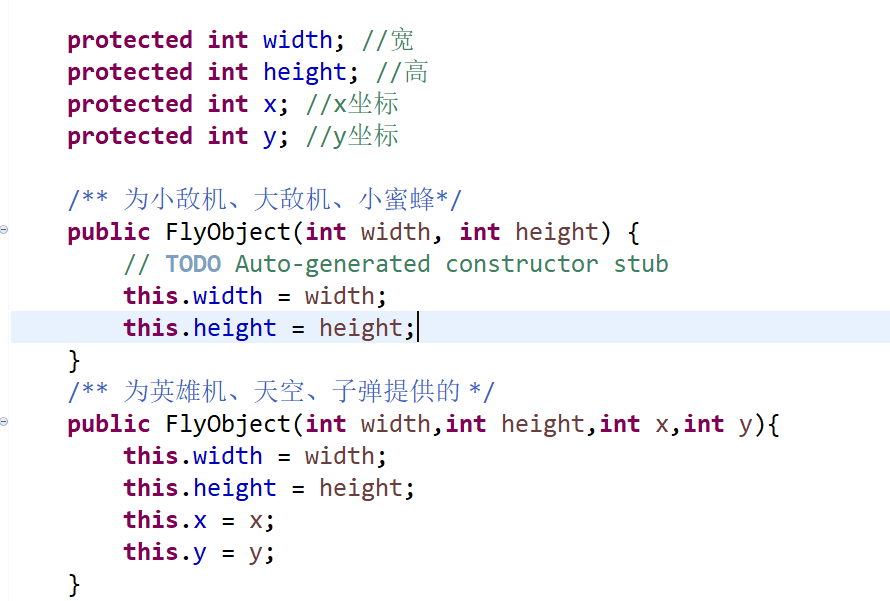
1. 设计小敌机、大敌机、小蜜蜂、子弹数组



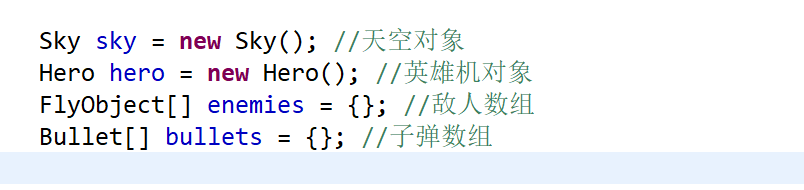
1. 创建父类FlyingObject，创建构造方法，整合数组
2. 创建FlyingObject超类，设计6个对象继承超类



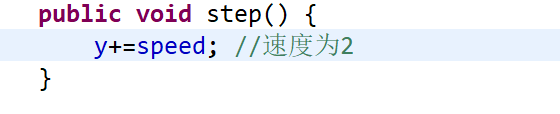
1. 在FlyingObject超类设计两个构造方法，6个对象类分别调用



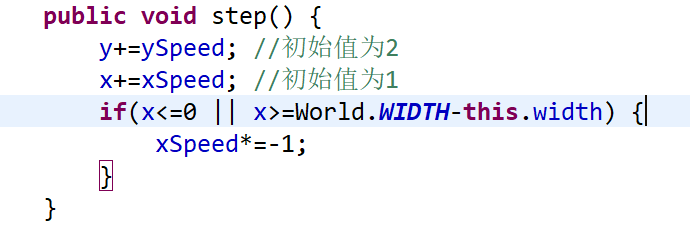
1. 将小敌机数组、大敌机数组、小蜜蜂数组合为FlyingObject数组



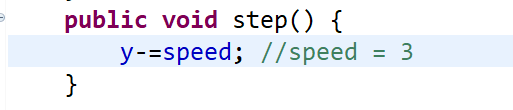
1. 6个类中重写step方法
2. 小敌机、大敌机的y坐标向下移动了:speed



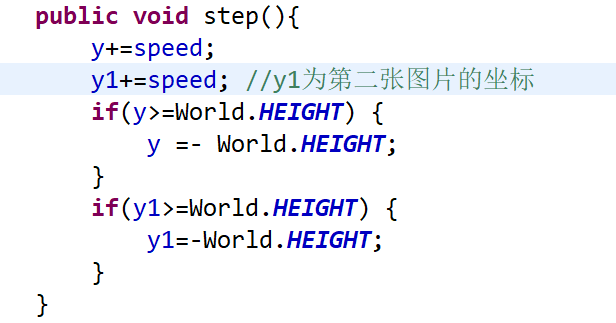
1. 小蜜蜂的x坐标移动了:xSpeed,y坐标向下移动了:ySpeed



1. 子弹的y坐标向上移动了:speed



1. 天空的y坐标和y1坐标向下移动了:speed



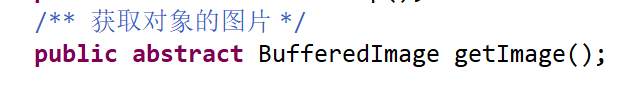
1. 英雄机切换图片啦!
2. 添加图片工具类，设置大小常量



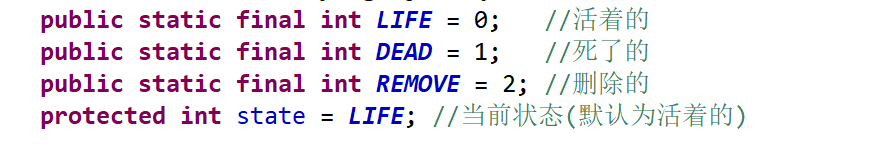




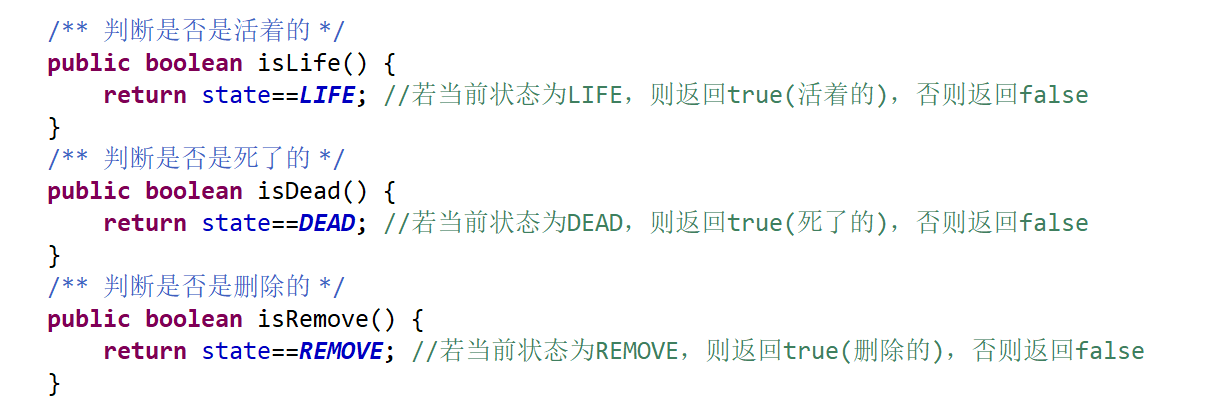
1. 设计抽象方法getImageh获取图像
2. 在FlyingObject中设计抽象方法getImage()用于获取对象的图片



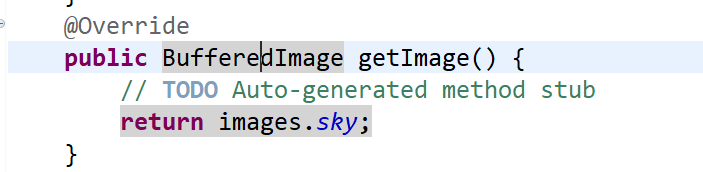
1. 在FlyingObject中设计LIFE活着的、DEAD死了的、REMOVE删除的常量，state变量表示当前状态



1. 在FlyingObject中设计isLife()、isDead()、isRemove()



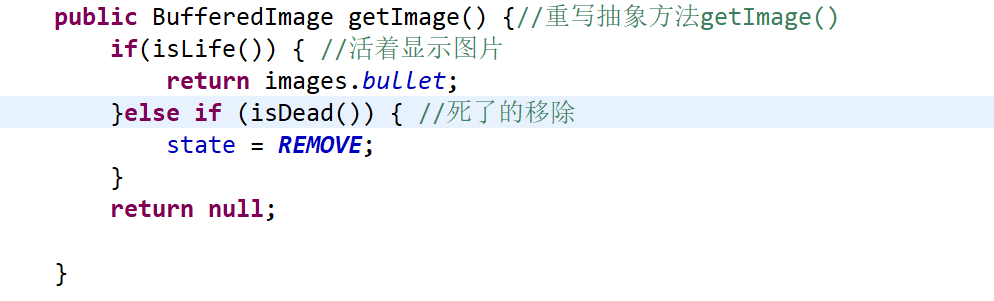
1. 重写抽象方法getImage():
2. 天空Sky，直接返回sky即可



2)子弹Bullet:

2.1)若活着的，直接返回bullet即可

2.2)若死了的，直接删除



3)英雄机Hero:

3.1)若活着的，返回两张图片来回切换



4)小敌机Airplane:

4.1)若活着的，直接返回第1张图片即可

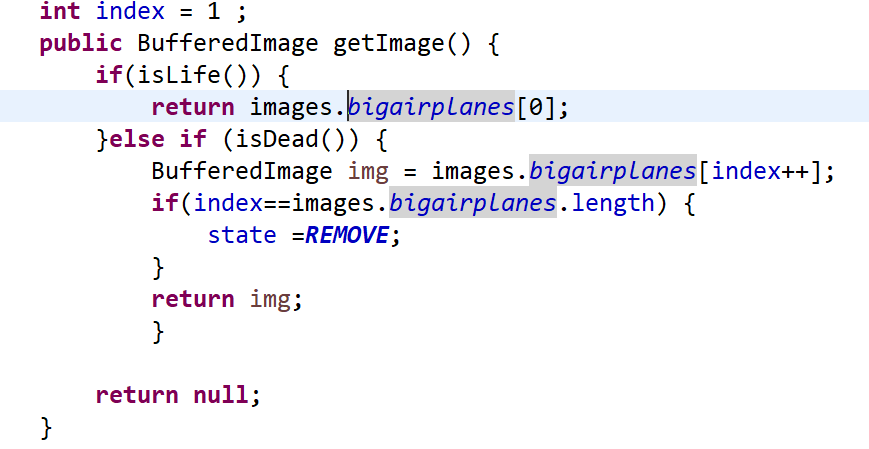
4.2)若死了的，返回第2张到第5张图片轮换，5后删除



5)大敌机BigAirplane:

5.1)若活着的，直接返回第1张图片即可

5.2)若死了的，返回第2张到第5张图片轮换，5后删除



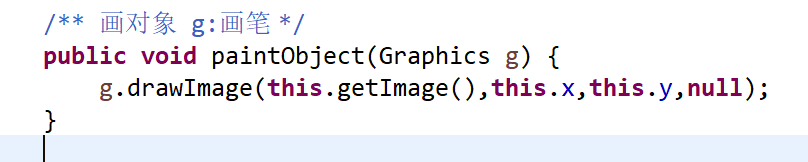
6)小蜜蜂Bee:

6.1)若活着的，直接返回第1张图片即可

6.2)若死了的，返回第2张到第5张图片轮换，5后删除

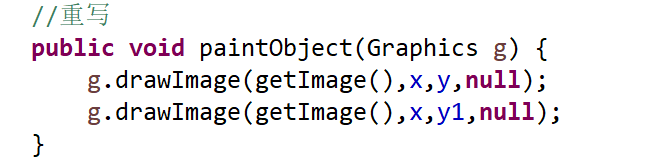


1. 在FlyingObject中设计paintObject()画对象



1. 在Sky中重写paintObject()画对象

天空的图片需要画2次(y坐标不同)



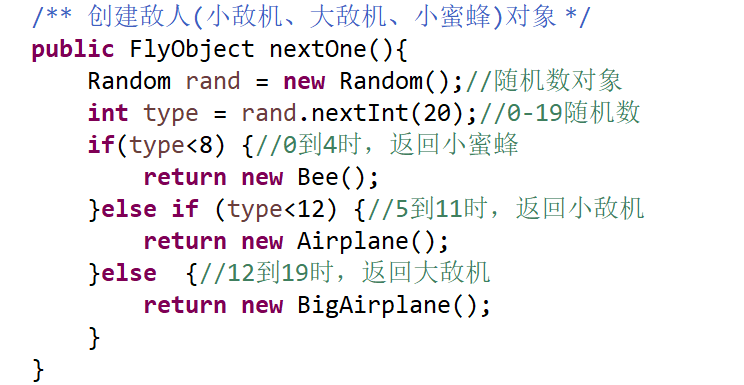
1. 在World类中重写paint()方法，调用paintObject()实现画对象



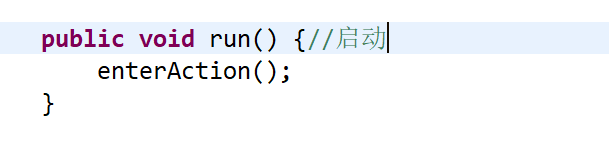
1. 设计入场及移动

1. 敌人入场:

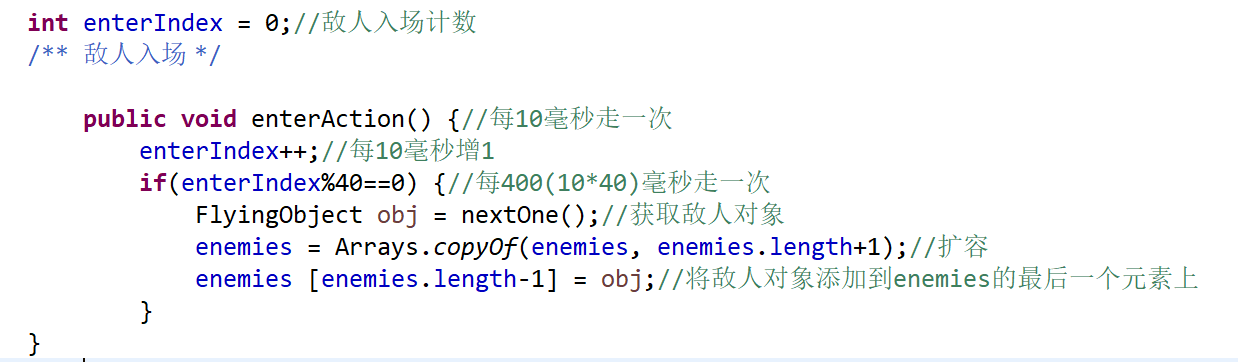
1) 敌人为窗口产生的，所以在World中设计nextOne()生成敌人对象



1. 敌人入场为定时发生的，所以在run()中调用enterAction()实现敌人入场

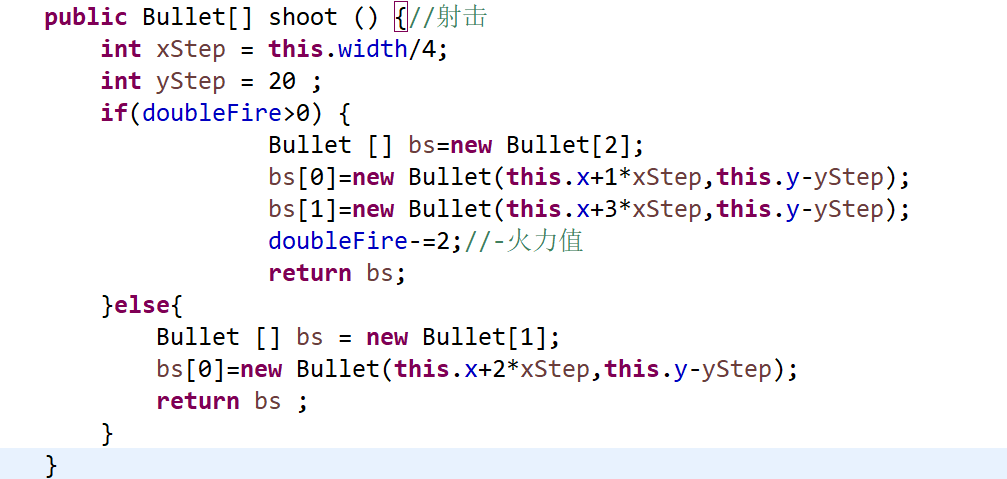


1. 在enterAction中:每400毫秒，获取敌人对象，enemies扩容，将对象放在最后一个元素上



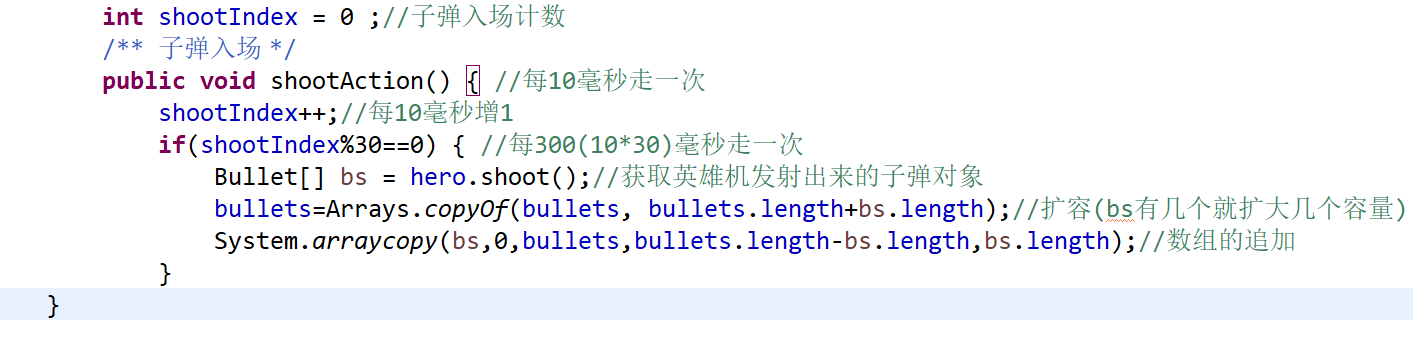
2.子弹入场:

1)子弹为英雄机发射出来的，所以在Hero中设计shoot()生成子弹对象



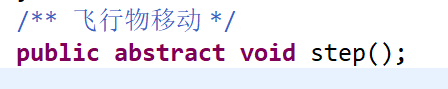
2)子弹入场为定时发生的，所以在run()中调用shootAction()实现子弹入场

3)在shootAction中:每300毫秒，获取子弹数组对象，bullets扩容，数组的追加



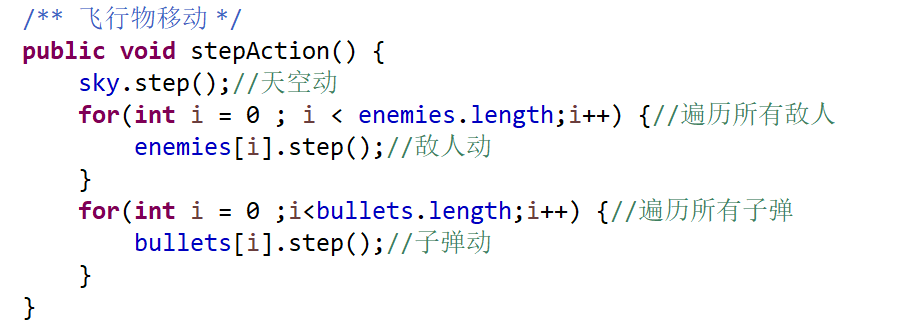
3.飞行物移动:

1)飞行物移动为所有派生类所共有的行为，所以设计在FlyingObject设计step()，派生类移动的具体方式不同，所以设计为抽象方法，派生类分别重写



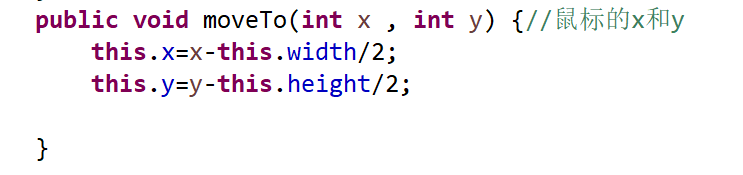
2)飞行物移动为定时发生的，所以在run()中调用stepAction()实现飞行物移动

3)在stepAction中:天空动，遍历所有敌人敌人动，遍历所有子弹子弹动



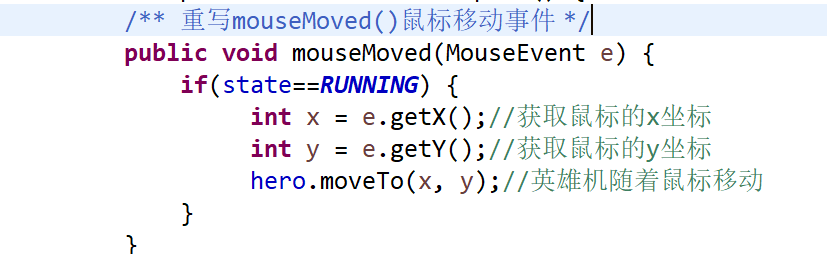
4.英雄机随着鼠标移动:

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



2)英雄机移动为事件触发的，所以在侦听器中重写mouseMoved()鼠标移动行为

3)在mouseMoved()中:获取鼠标的x和y坐标，调用Hero的moveTo()实现英雄机移动



5.删除越界的敌人和子弹:

1)在FlyingObject中设计outOfBounds()实现检测敌人的越界,在Bullet中重写outOfBounds()实现检测子弹的越界



2)删除越界敌人和子弹为定时发生的，所以在run()中调outOfBoundsAction()实现删除越界敌人和子弹

3)在outOfBoundsAction()中:

声明不越界敌人/子弹数组，遍历敌人/子弹数组，

判断:若不越界，则将不越界敌人/子弹添加到不越界敌人/子弹数组中，

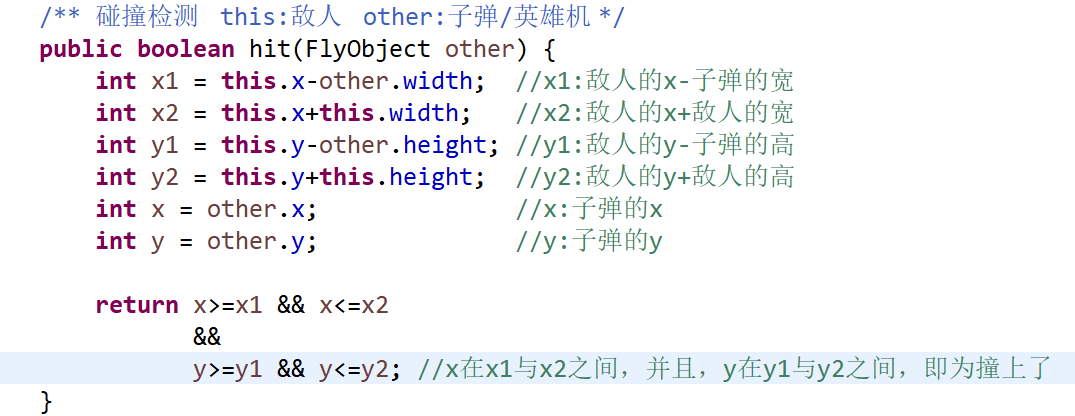
最后:将不越界敌人/子弹数组复制到enemies/bullets中



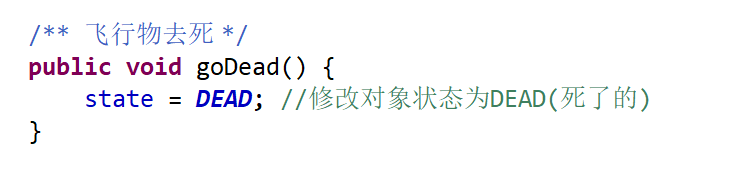
1. 子弹与敌人碰撞与命的计算

1.子弹与敌人的碰撞:

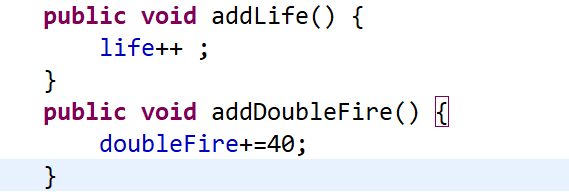
1)在FlyingObject中设计hit()实现检测敌人与子弹的碰撞



2)在FlyingObject中设计goDead()实现飞行物去死



3)在Hero中设计addLife()增命、addDoubleFire()增火力



4)子弹与敌人的碰撞为定时发生的，所以在run()中调用bulletBangAction()实现子弹与敌人的碰撞

在bulletBangAction中:

遍历子弹得子弹，遍历敌人得敌人，

判断若子弹活着并且敌人活着并且撞上了:

子弹去死、敌人去死

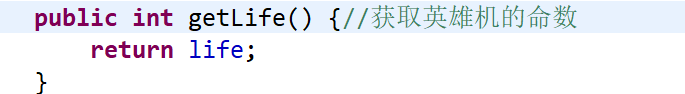
判断若被撞敌人为Enemy分，则强转为分，玩家得分

若被撞敌人为Award奖励，则强转为奖励，英雄机得奖励

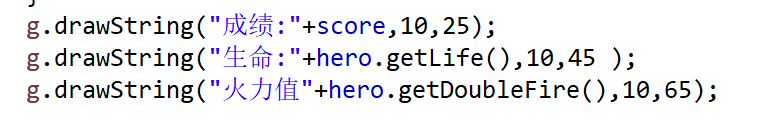


2.画分和画命:

1)在Hero中设计getLife()获取英雄机的命数



2)在World类的paint()中: 画分和画命

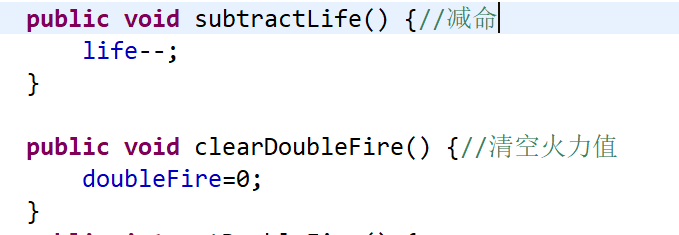


1. 英雄机与敌人的碰撞与状态

1.英雄机与敌人的碰撞:

1)借用FlyingObject中hit()碰撞检测、goDead()飞行物去死

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

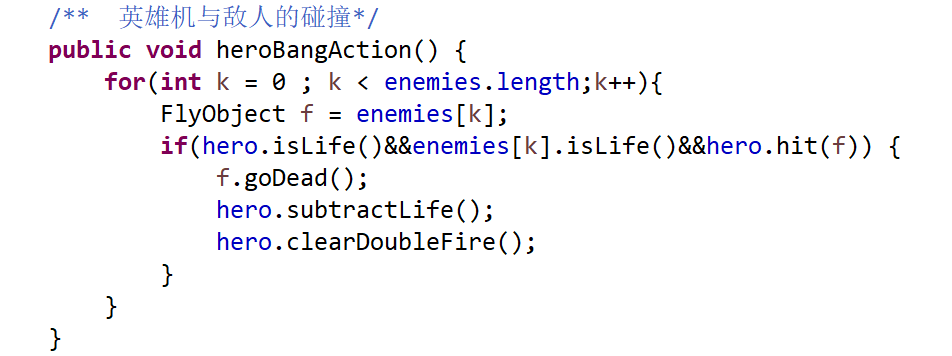


2)英雄机与敌人的碰撞为定时发生的，所以在run中调用heroBangAction()实现英雄机与敌人的碰撞

在heroBangAction()中:

遍历所有敌人得敌人，判断若都活着的并且撞上了:

敌人去死、英雄机减命、英雄机清空火力值



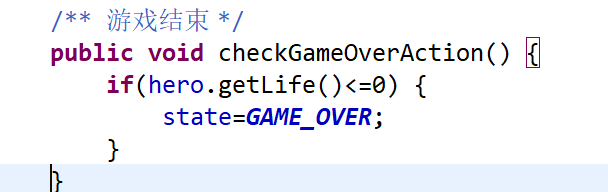
2.检测游戏结束:

1)借用Hero中的getLife()获取命数

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

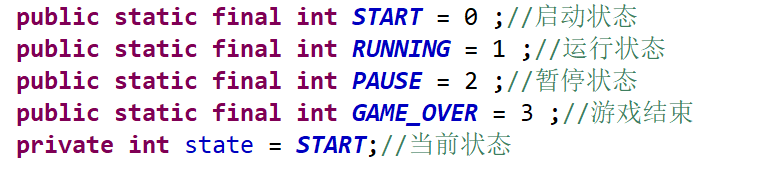
在checkGameOverAction中:

判断若英雄机的命数<=0，刚将当前状态修改为游戏结束状态



3.画状态:

1)在World中设计4个状态常量，state变量表示当前状态

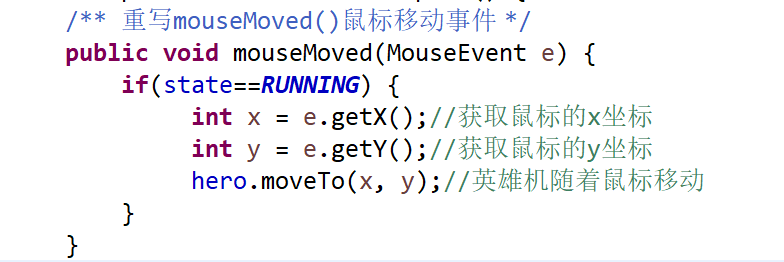


在Images中设计3张状态图片

在World的paint()方法中:设计不同状态下画不同的图

2)在run()中设计那一堆action只在运行状态下执行

在mouseMoved()中设计英雄机移动只在运行状态下执行



3)重写mouseClicked()鼠标点击事件:

启动状态变运行状态:游戏结束状态先清理现场再变启动状态

重写mouseExited()鼠标移出事件:运行变暂停

重写mouseEntered()鼠标移入事件:暂停变运行

