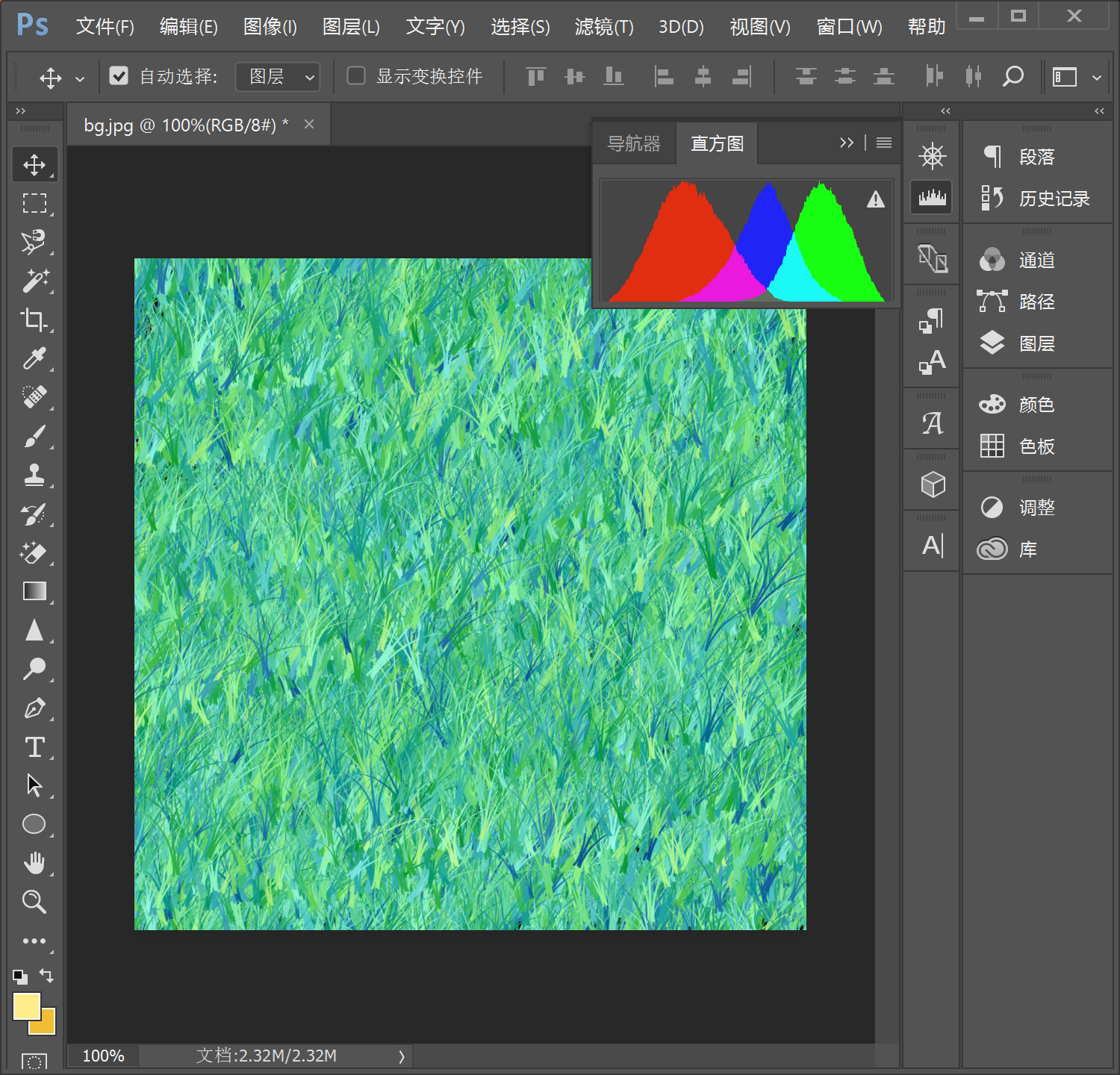
**系统界面设计**

### 1.游戏的背景界面设计

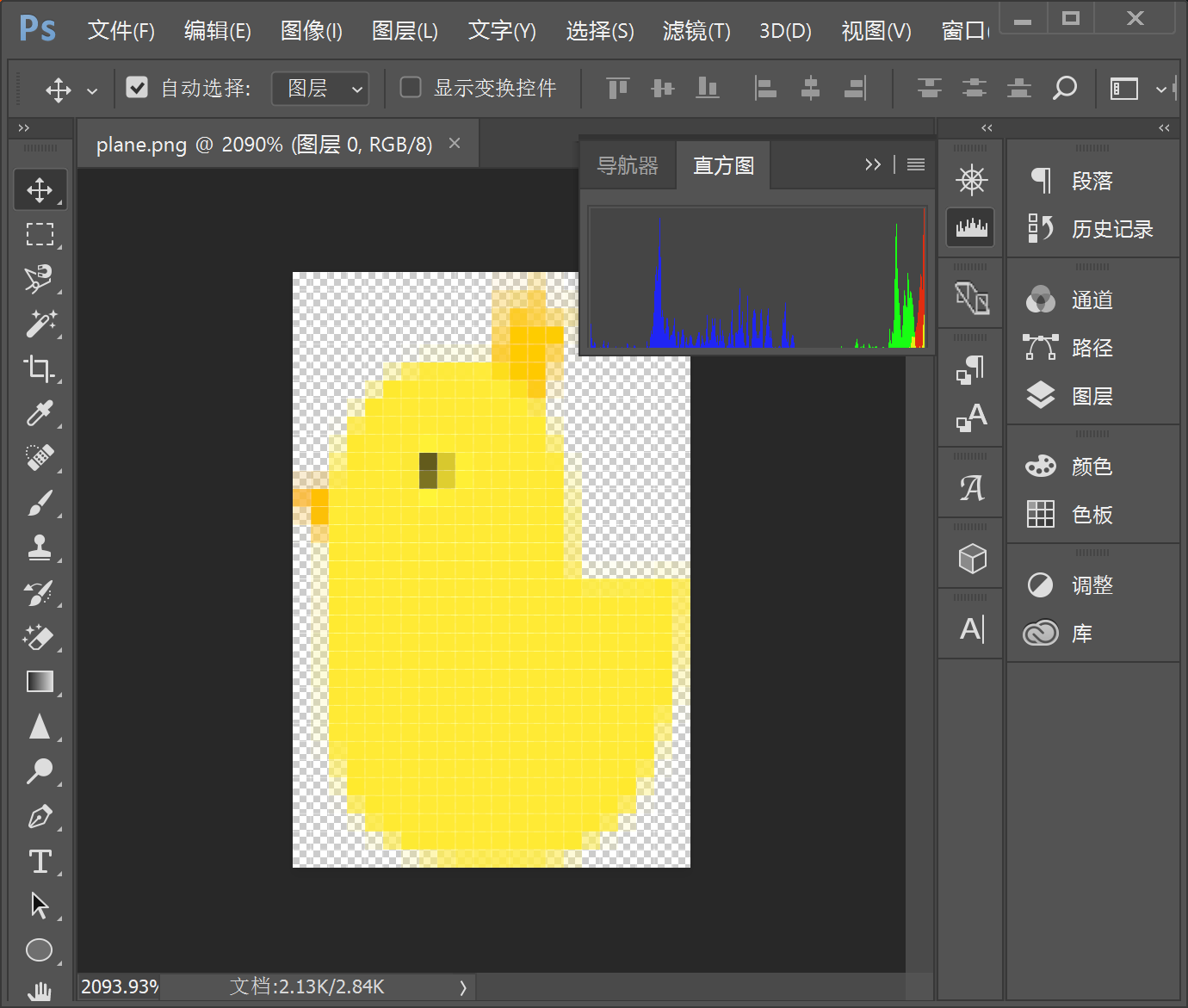
游戏的背景界面是在Photoshop中用草形草笔选择青色和蓝色画出，图像大小为900\*900像素，图片画好后存为jpg格式。

当游戏开始启动时，程序会自动读取背景图片，将背景图片展现在主面板之上。



### 2.游戏中的小鸡形象的设计

游戏中小鸡形象的设计主要使用了软件PhotoShop，在空白面板上，先新建一个宽22像素、高33像素的画布，然后将画布的设置为不透明度0%，再画出小鸡的图形，然后存为png格式。该形象的设计主要是为了方便后续小鸡与麻雀碰撞的检测需要。

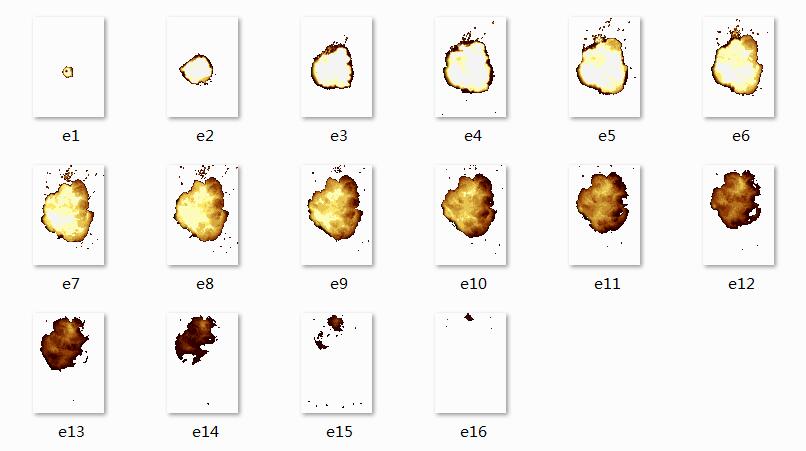
plane

### 3.爆炸效果的设计

游戏中在小鸡被麻雀捉到后，需要出现一个爆炸效果.

我们定义Explode类来表示爆炸的信息，爆炸类和普通类不一样的地方在于它实际上存储了一系列爆炸的图片，然后进行轮播。

我们这里准备了一系列爆炸图片：



在设计过程中，我们定义Image[]来保存图片信息，并且使用static代码块，也就是在类加载时就加载这些图片，并且从属于类，通过此方法，我们不需要每次创建爆炸对象都加载图片，这样保证了游戏运行的效率。

static {

for (int i = 0; i < 16; i++) {

imgs[i] = GameUtil.getImage("images/explode/e" + (i + 1) + ".gif");

imgs[i].getWidth(null);

}

}

同时，我们通过使用计数器count来控制到底画哪个图片，由于图片的命名规范，是按照顺序从1-16，这样程序只需依次读取这些图片对象即可。

public void draw(Graphics g) {

if (count <= 15) {

g.drawImage(imgs[count], (int) x, (int) y, null);

count++;

}

}

帧动画是通过一定顺序来加载之前准备好的静态图片来实现动画的视觉效果，本游戏的爆炸效果就是通过这种方式实现。

//小鸡和炸弹的接触检测！！！

boolean peng = shells[i].getRect().intersects(plane.getRect());

if(peng){

plane.live = false;

if(bao ==null){

bao = new Explode(plane.x, plane.y);

endTime = new Date();

period = (int)((endTime.getTime()-startTime.getTime())/1000);

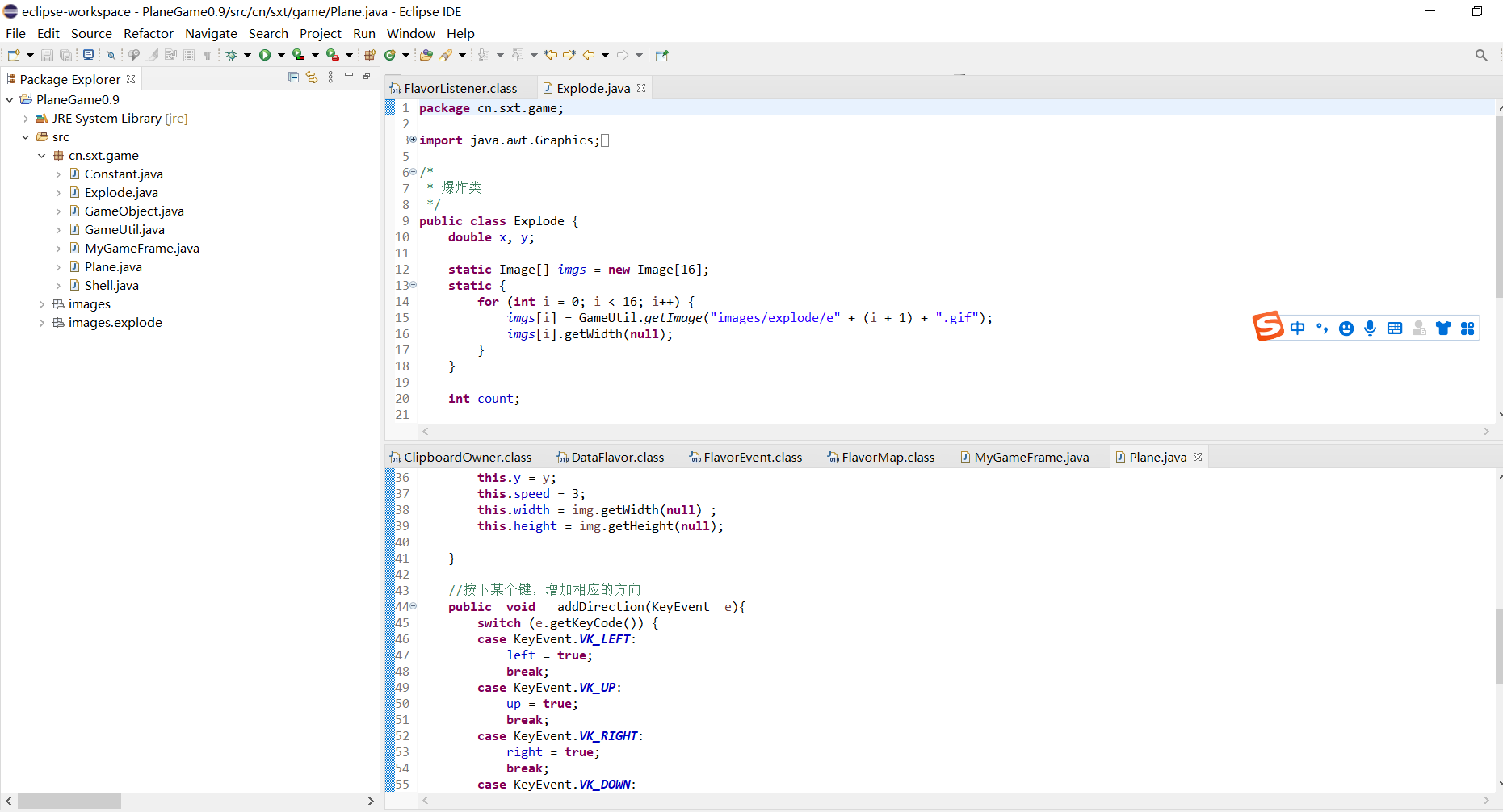
}

bao.draw(g);

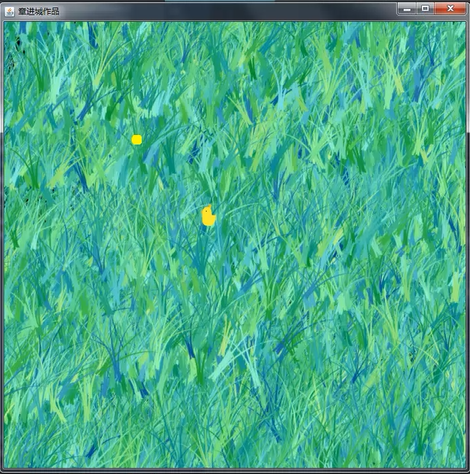
}

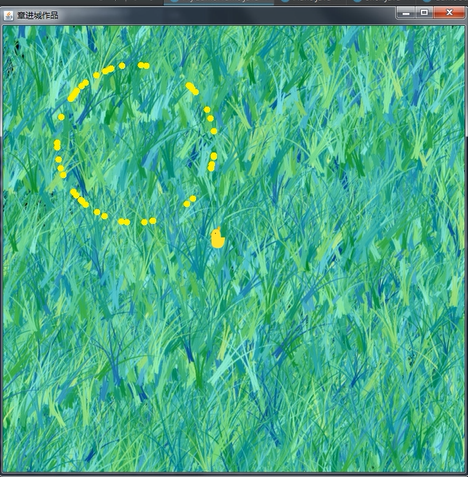
## 游戏界面效果展示

在eclipse中打开游戏界面

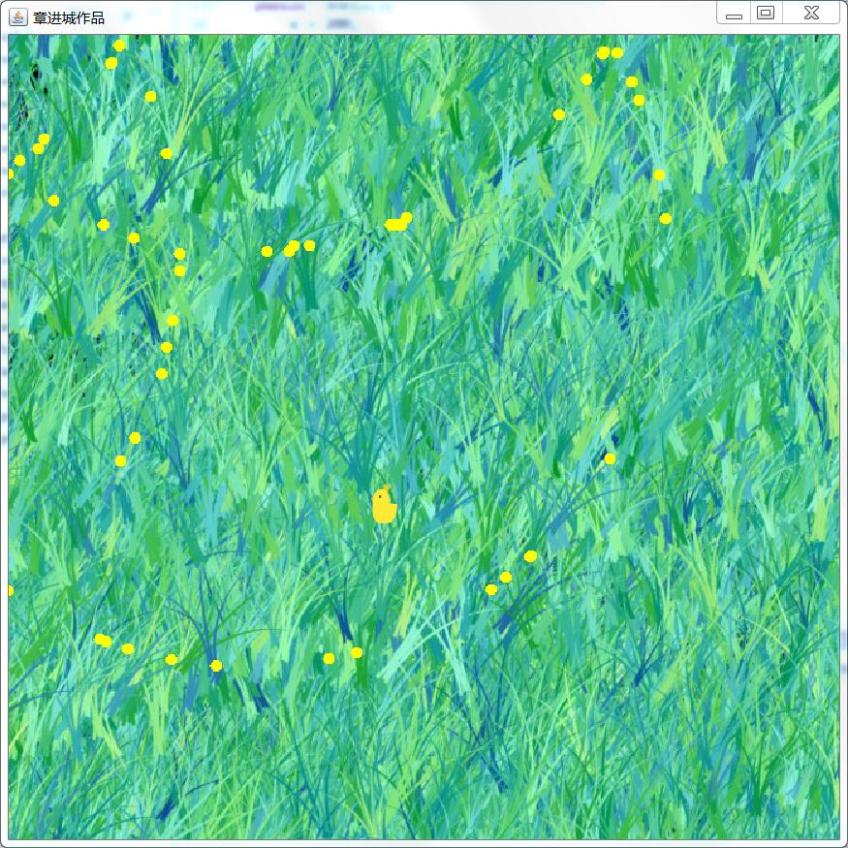


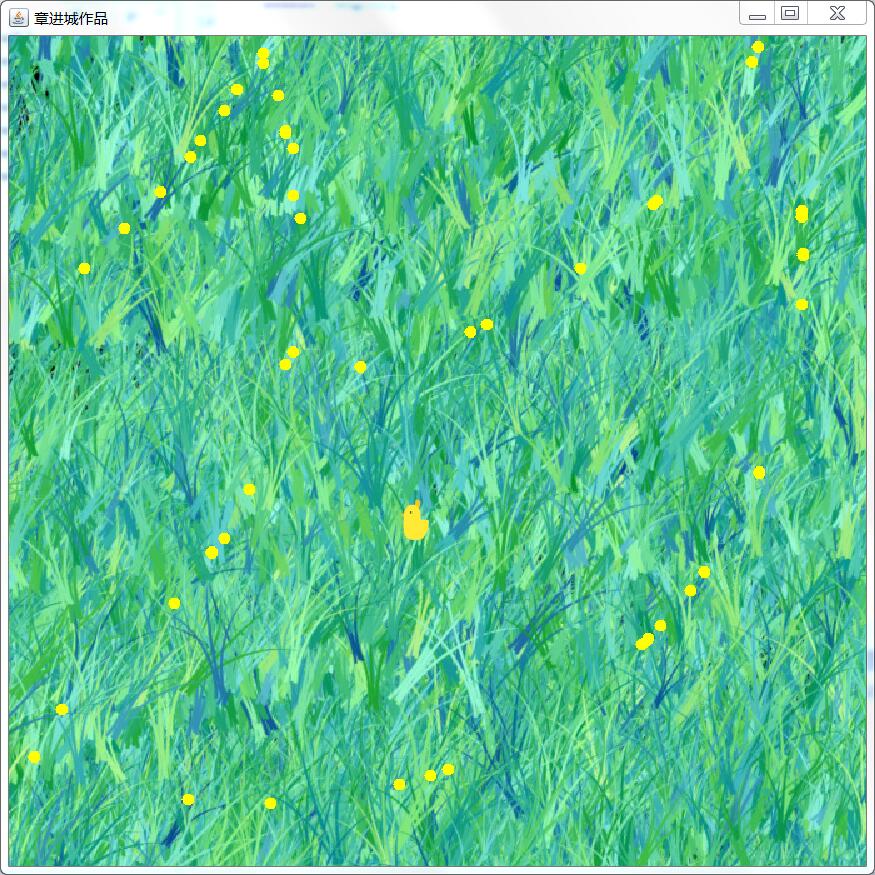
### 5.小鸡躲炸弹游戏运行界面

小鸡躲的游戏运行效果图如下所示： 



炸弹随机运动，四散分布在整个面板的周围。当炸弹运动到面板的边缘时，则反弹继续在面板中移动





小鸡躲炸弹游戏运行页面中间的小鸡由玩家控制其进行上下左右移动来躲避由界面200\*200像素处生成的自由移动的炸弹的捕捉。

**6.小鸡躲炸弹游戏结束界面**

当游戏中的小鸡被炸弹捉住时游戏结束，在游戏界面中显示经过多少秒小鸡被炸弹炸到。此时页面中小鸡消失。

在被捉住的瞬间，小鸡爆炸

