1.加载一个窗口

1>>首先我们需要的继承一个类叫 Frame

**public** **class** ShiYan **extends** Frame{ //GUI 编程中 不擅长写桌

}

2>>提供一个方法将窗口画出

**public** **class** ShiYan **extends** Frame{ //GUI 编程中 不擅长写桌面软件

//加载一个窗口

**public** **void** launchFrame(){

setSize(500,500); //窗口的大小

setLocation(100,100); //窗口的位置

setVisible(**true**); //默认是不可见的

}

}

3.main函数将创建窗口的方法调入

**public** **static** **void** main(String args[]){

ShiYan a = **new** ShiYan();

a.launchFrame();

4.现在我们的窗口 能打开了 但是不可以 关闭，现在开始 研究关闭

//现在的窗口打开，但是不可关闭，现在我们开始研究关闭

addWindowListener(**new** WindowAdapter(){

**public** **void** windowClosing(WindowEvent e){

System.*exit*(0);

}

});

这是一个调用的方法中 使用了一个 重写了子类的一个方法

5.现在我们开始学习画画了（此时我们需要重写一个方法 叫做paint ）

//窗口现在是有了，现在我们需要在里面画东西了（但是我们此时我们需要重写一些东西）（**不需要调用也会执行**）

@Override

**public** **void** paint(Graphics g){ //这个里面的东习就像画笔一样

g.drawLine(100, 200, 200, 300);

//第一点的坐标，第二点的坐标 （100,200）（200,300）

g.drawRect(100,200,100,100); 矩形

g.drawOval(100,200,100,100); // 画圆 其实就是 画了一个矩形的内切圆

//椭圆 其实就是 长度和宽度不一样就可以了



5.2上面的是画实线现在我们开始填充颜色，当然默认是黑色的

g.setColor(Color.*red*); //这个事填充颜色的工具

g.fillRect(50,50,50,50);

g.fillOval(300, 300, 100, 100);

5.3基本上是画出来了，画笔不是一个画，所以我们用完之后要洗一下

初始的画笔是Color a = g.getColor();

5.2中写的要写入这里

最终的画笔是 g.setColor(a);

5.4现在来研究一下 写入得文字

Font f = **new** Font("楷体",Font.*BOLD*,20);定义文字

g.setFont(f); 写入文字

g.drawString("这里是张宇晋画的一个字符串",100,250);

5.6现在开始导入一张图片（下面这个是一个图片加载工具，之后我们才会开始加载，我们现在还不会学习下面的东西）

**public** **class** JiazaiTupian {

**public** **static** Image getmage(String path){

URL u = JiazaiTupian.**class**.getClassLoader().getResource(path);

BufferedImage img = **null**;

**try** {

img = ImageIO.*read*(u);

} **catch** (IOException e) {

// **TODO** 自动生成的 catch 块

e.printStackTrace();

}

**return** img;

}

//现在图片 基本上已经写完了 现在我们开始使用它

}

5.7现在开始导入一个图片

Image haha = JiazaiTupian.*getmage*("images/1340270956\_-1719592020\_16544\_sProdImgNo\_3.jpg");

下文的paint中是 g.drawImage(haha,100,100,**null**);

5.8这里图片加载基本上是完了，记得我的这里是使用的一个，图片加载工具，我们的这个图片加载工具，中static是可以随便调用的，但是，我们需要设置一个 构造器使它变成private

**public** **class** JiazaiTupian {

**private** JiazaiTupian(){}

6. 动画的实现

1.重画

**class** Donghua **extends** Thread{

//重写方法

**public** **void** run(){

**while**(**true**){

repaint(); 这里就是我们已经画好的东西，在开始画画了

**try** {

Thread.*sleep*(40); //这是我们多长是假重新画一次

} **catch** (InterruptedException e) {

// **TODO** 自动生成的 catch 块

e.printStackTrace();

}

}

}

}

2.列出窗口的地方进行加载

**public** **void** launchFrame(){

setSize(500,500); //窗口的大小

setLocation(100,100); //窗口的位置

setVisible(**true**); //默认是不可见的

**new** Donghua().start(); //现在这里是重画

这时我们已经可以重新画画，但是我们还需要对它进行动态

**private** **double** x = 100,y = 200;

g.drawImage(haha,(**int**)x,(**int**)y,**null**); 要进行强制转化

x += 3; 在画画过程中我们可以对他进行加一个数，然后就是画画了吧！

y += 4;

7.创建一个动画的项目，是他按照气泡一样的运动（在右下角的截图中有序的运动）

**class** huahua **extends** Thread{

**public** **void** run(){

**while**(**true**){

repaint();

**try**{

Thread.*sleep*(1);

}**catch**(InterruptedException e){

e.printStackTrace();

}

}

}

}

**public** **static** **void** main(String args[]){

Dongqilai d = **new** Dongqilai();

d.zuoyoutan();

}

}

**public** **class** Dongqilai **extends** Frame {

Image wo = JiazaiTupian.*getmage*("images/1340270956\_-1719592020\_16544\_sProdImgNo\_3.jpg");

**public** **void** zuoyoutan(){

setSize(500,500);

setLocation(100,100);

setVisible(**true**);

**new** huahua().start();

addWindowListener(**new** WindowAdapter(){

**public** **void** windowClosing(WindowEvent e){

System.*exit*(0);

}

});

}

**double** i = 100,j = 200,drirectx = 1,drirecty = 1;

**public** **void** paint(Graphics g){

g.fillRect((**int**)i,(**int**)j,20,20);

**if**(i >= 480){

drirectx = -1;

}

**if**(i <= 0){

drirectx = 1;

}

**if**(j >=480){

drirecty = -1;

}

**if**(j <= 20){

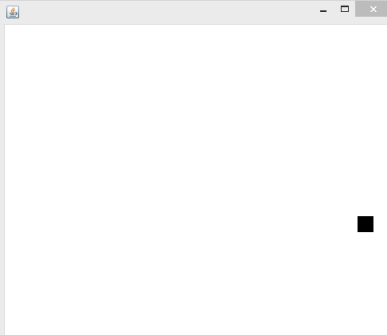
drirecty = 1;

}

i = i + drirectx;

j = j + drirecty ;

}



8.创建一个 台球的 运动（可以控制停止与启动），我们可以按照上面到的东西来制作



**private** **double** x = 100,y = 100;

**private** **double** degree; //这是个弧度 【0,2π】

**public** **void** paint(Graphics g){

g.fillOval((**int**)x,(**int**)y,30,30);

x += 3 \*Math.*cos*(degree);

y += 3 \*Math.*sin*(degree);

}

这里使用的degree 是一个弧度 上面这个就是代表的

8.2当我们的球在下面的时候我们会一直以 像右和像下的坐标为正，所以现在 离那个

1》》弹到了下面的桌子上时 为 - degree



2》》弹到了上面的桌子上时 为



|  |
| --- |
|  |



3》》弹到了右面的时候为 π-degree

4》》弹到了左面的时候为 π-degree

**注意 这里的π 写为 Math.PI,Math.cos()**

**public** **void** paint(Graphics g){

g.fillOval((**int**)x,(**int**)y,20,20);

x += 10 \* Math.*cos*(degree);

y += 10 \* Math.*sin*(degree);

**if**(y > 500 - 20||y < 20){

degree = - degree;

}

**if**(x < 0||x > 500 - 20){

degree = Math.*PI* - degree;

}

8.3让台球停下了（将上面的 10 变成speed，用一个if条件让他停下来）

**double** speed = 10;

**public** **void** paint(Graphics g){

g.fillOval((**int**)x,(**int**)y,20,20);

x += speed \* Math.*cos*(degree);

y += speed \* Math.*sin*(degree);

**if**(y > 500 - 20||y < 20){

degree = - degree;

}

**if**(x < 0||x > 500 - 20){

degree = Math.*PI* - degree;

}

**if**(speed > 0){

speed -=0.05;

}**else**{

speed =0;

}