推箱子开发流程：

窗口

背景

障碍

人物

箱子

笼子

让人物动起来

箱子是如何被动移动？

游戏胜利判定规则？

要不要进行下一局？

要不要显示游戏胜利结果？

--------------------------

界面制作：awt+swing

窗口制作：

1.创建一个MainFrame类

2.继承Frame extends Frame(写在MainFrame后面，导入这个类)

3.写构造方法 public MainFrame() (写在类的大括号内部)

4.在构造方法内对窗体进行各种设置

4.1设置了标题 this.setTitle("你的标题");

4.2设置了位置和大小 this.setBounds(a,b,c,d); 距离屏幕左上角位置、c长d宽

4.3设置是否显示 this.setVisible(true); 设置显示状态为显示

-------------------------

背景制作：

背景是一张图片，如何放入到窗体中？

需要借助一个组件，该组件可以显示图片。

Swing包的JLabel包

1.创建一个图片对象，该对象需要一张图片，并将该图片放到项目目录下，图片的名称以字符串的形式描述

Icon i = new ImageIcon("bg.png");

2.使用可以制作背景的JLabel组件，将图片对象加入进去

JLabel lab\_bg = new JLabel(i);

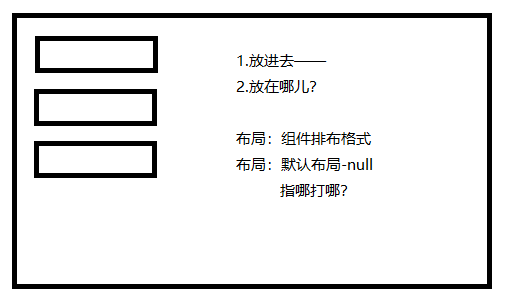
3.设置这个背景组件大小和位置

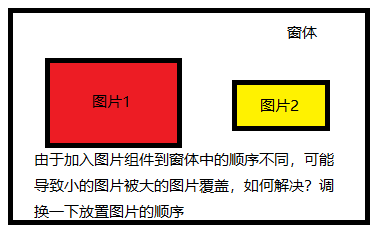
lab\_bg.setBounds(12,36,600,495);

如何把我要做的东西添加到窗体里面？

1.什么东西?有东西才能添加

2.怎么添加？





推箱子人物（灰太狼）制作及人物移动控制：

人物制作

1.人物制作也是一张图片的结构，因此在制作时可以像制作背景图片一样的方式制作

2.为了使人物的制作独立处理，需要设置一个单独的方法来完成

Private void wolfInit ()

3.制作过程同背景制作过程

3.1制作一个图片的对象，使用的素材图片最好是透明背景的，并且尺寸要提前计算好

Icon i = new ImageIcon("wolf-zm.png");

3.2使用一个放置图片的组件，将图片放进去

JLabel lab\_wolf = new JLabel(i);

3.3设置该组件的显示位置和大小，大小要参看原始图片的尺寸，当前位置是随意设置的，大小是根据图片的大小设置的

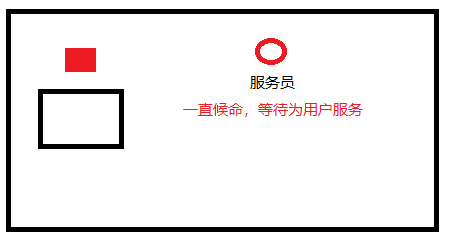
lab\_wolf.setBounds(300,300,122,151);

3.4将组件添加到窗体中

This.add(lab\_wolf);

注意事项：由于制作多个图片组件放入到窗体中时存在放入的顺序，先放入的在最上面，后放入的在最下面，，所以要根据需要调整该位置

2.事件监听机制



要到达以下目标：

点击键盘上的右键（→），想让人物向右移动一下

1.如何让JAVA语言知道我们点击键盘了？

需要使用JAVA中的事件监听机制

a.为窗口添加一个可以监督用户是否点击键盘的监听器 this.addKeyListener(this);

上述语句可以写在窗口初始化的后面

b.上述语句添加完毕后，会出现错误，需要使当前类实现一个监听接口。

public class MainFrame extends Frame implements KeyListener{}

c.上述操作执行完毕后，又会出现一个错误，需要添加接口未实现的方法。原因：一个类实现了一个接口后，要实现这个接口中所有的抽象方法。

public void keyPressed(KeyEvent e){

//TODO Auto-generated method stub

}

public void keyReleased(KeyEvent e){

//TODO Auto-generated method stub

}

public void keyTyped(KeyEvent e){

//TODO Auto-generated method stub

}

d.在下面的方法中添加一句话，测试一下，是不是运行？如果程序运行后，点击键盘能够看到显示文字，则键盘控制成功。

public void keyPressed(KeyEvent e){

System.out.println(“测试一下”);

}

2.如何知道我们点击键盘上面的右键，而不是其他键？

JAVA语言中提供了一个东西，键码值，通过输入的按键不同，得到的键码值不同

左37 上38 右39 下40

通过键码值分辨出你点击的是哪个按键

int key =e.getKeyCode();

3.人物移动控制

点击了右键后，如何让人物动起来？

a.首先获取一下人物当前的位置

lab\_wolf.getLocation()

位置获取后，里面有一个横坐标x值，纵坐标y值

int x = (int) lab\_wolf.getLocation().getX();

int y = (int) lab\_wolf.getLocation().getY();

b.设置人物的位置，使用前面设置窗口的操作方式来进行

lab\_wolf.setLocation(x+”图片长值”,y)

上面为向右移动

如果向左移动，横向-“图片长值”，纵向不变

如果向上移动，纵向-“图片宽值”，横向不变

如果向下移动，纵向+“图片宽值”，横向不变

c.人物移动后要对人物的图片进行更新

1.设置要更新的图片内容

Icon i = new ImageIcon("htl-zm.png");

2.将该图片设置到对应的组件上面

lab\_wolf.setIcon(i);

---------------------------------------------------------------

推箱子箱子（懒洋洋）制作、目的地（笼子）制作及障碍（树木）制作

整体方案模型设计

1.人物及其所有东西放的位置一定要将窗体边缘的距离考虑进去

目前我的窗体中设置背景的位置是距离左边12，距离上边36，所以所有的组件再添加时，必须要加上这个修正值，如果不加就出现了无法对齐的现象。

横方向修正：12像素

纵方向修正：36像素

推箱子人物（灰太狼）位置调整

原始设计人物默认出现的位置在：

Lab\_wolf.setBounds(300,300,50,50);

经过修正后人物应该出现在下列位置：

Lab\_wolf.setBounds(312,336,50,50);

推箱子人物（灰太狼）移动控制调整

移动控制对应的坐标位置由于是动态获取的，因此不需要修改

推箱子箱子（懒洋洋）制作

按照人物的制作方式就可以制作出箱子的模型，这里仅仅图片不一样

1.创建图片，图片是羊的图片

Icon i = new ImageIcon("sheep-no.png");

2.使用JLabel组件模拟箱子模型

JLabel lab\_sheep1 = new Jlabel(i);

3.对箱子的位置进行设置，无论设置在什么位置，千万不能重复（姜片一两考虑进去）

Lab\_sheep1.setBounds(312,236,50,50)；

4.箱子制作完毕后，将其添加到窗体中

this.add(lab\_sheep1);

5.由于最终游戏中有3个箱子，因此要对上述过程重复制作

5.1图片熟不需要重新制作，因为所有箱子使用的是同样的图片

5.2多个箱子制作过程几乎完全相同，但是要使用不同的名称

Lab\_sheep1 Lab\_sheep2 Lab\_sheep3

5.3多个箱子为了显示时不重叠，需要使用不同的坐标，同时不要与其他的东西重叠

推箱子目的地（笼子）制作

笼子的制作于羊的制作完全相同

JLabel lab\_target2 = new JLabel(i);

lab\_target2.setBounds(650+12,300+36,50,50);

this.add(lab\_target2);

注意：由于最终箱子对应的显示效果应该是在笼子里（后）面的，因此在组件添加时一定要考虑让箱子的图片显示在羊的上面，所以我们要对其进行位置的调整

1.先设定笼子

2.再设羊和狼

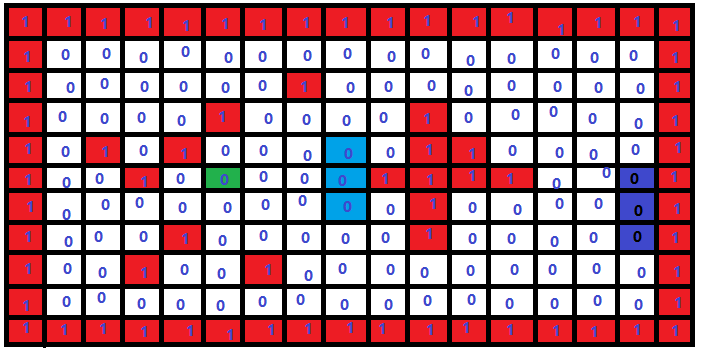
3.最后设定背景

推箱子障碍（树木）制作

障碍制作方式分析：

将障碍转换成一组数据，好多行数据，1的数据代表障碍，0的数据代表空地

障碍模型设计：



上述模型实际上与JAVA语言中的二维数组模型完全一样

横着做多少？列

竖着做多少？行

背景图片多大？800\*600

每一个现实的内容多大？50\*50

横向可以显示的图片总数是：800/50=16个

纵向可以显示的图片总数是：600/50=12个

即12行16列

需要使用二维数组来模拟上述数据，12行16列

下面这个数组定义在class的大括号内

int[][] datas = {

{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},

{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1},

{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1},

{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1},

{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1},

{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1},

{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1},

{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1},

{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1},

{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1},

{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1},

{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1}

};

遍历数组将所有值为1的位置添加障碍物

Icon ic = new ImageIcon("tree.png");

//遍历二维数组

for (int i = 0;i <= 11;i++){

for (int j = 0;j <= 15;j++){

if (datas[i][j] == 1) {

//障碍的初始化

//2.创建JLabel

JLabel lab\_tree = new JLabel(ic);

//3.设置大小位置

lab\_tree.setBounds(12 + 50 \* j, 36 + 50 \* i , 50, 50);

//4.添加到窗体中

this.add(lab\_tree);

}

}

}

场景更换

动态更换数组中的0和1

动态场景更换（思考）

碰撞实验

当人物移动过程中，遇到了障碍，应该不能够通过障碍，此时我们就要考虑所有模型之间的碰撞问题，也就是我们所说的碰撞实验

模型碰撞思想

1.人物在屏幕上显示，显示在哪里？程序怎么知道人物在什么位置？

显示模型：车票售票大厅，售票人员：关心旅客有没有身份证，旅客有没有钱，旅客买票从哪里到哪里？

2.什么是障碍？计算机怎么知道数目就是障碍的？羊是不是障碍？

二维数组中的数据1代表障碍，0代表空地

我们设计数据2代表人物，人物的数据在原始的二维数组是不存在的，因此人物添加到程序中时，必须把这个数据2放到二维数组中

2.1代码应该写在什么位置？

应该把这个数据2的修改放入到人物创建的方法中

放在名称为wolfInit的方法中，也就是人物初始化的方法

2.2代码应该怎么写？

将数据2放到灰太狼对应的位置上面

思考：人物在移动的过程中，这个数据2的位置在不停的改变

我们想找一个地方，写上现在人物在哪？

设计一个值，用来描述人物现在在第几行

在数组的下面添加一个int wx;

设计一个值，用来面熟人物现在在第几列

在数组的下面添加一个int wy;

在人物初始化的第一句写上：

wx = 6;wy = 4;

把人物初始化的位置改成由x和y计算而来

lab\_wolf.setBounds(50\*wx+12,50\*wy+36,50,50);

人物的移动会导致这个wx与wy的变化，具体的变化规则是

移动控制坐标改变的代码写在对应的移动事件中

右移：wx=wx+1

左移：wx=wx-1

下移：wy=wy+1

上移：wy=wy-1

3.当人物遇到障碍后，怎么办？障碍针对人物的位置不同，是否会对人物的移动产生影响？

树木是障碍，但是这只是用户看到的结果，实际上树木在程序中以数字1的格式出现的

所谓的遇到障碍，就是判断一下我下面要移动过去的位置上面是不是1？

模型判断误区

datas[x][y]

x描述的是列，y描述的是行

控制人物遇到障碍后，不做任何事情，将下列代码

if(datas[wy+-1][wx+-]==1) {

return;

}

人物模型移动设计

人物模型移动是通过两个整数模拟当前人物在数组中的位置，同时也是将屏幕划分成若干个小方格，然后人物在这些小方格间跳动

碰撞模型移动设计

碰撞模型是通过数组中的数据进行检测的，不同的数据碰撞检测不同，

碰撞模型实现

对不可通过的数据进行判断，比如说树木对应的数据1，不可通过，那么就可以使用判断的形式提前结束移动操作

特殊碰撞实现

当人物遇到箱子（羊）怎么办？

箱子在程序中如何描述？怎么知道前面是箱子而不是空地？也不是树木？

使用数据4表示箱子，原始的数据中没有4，箱子数据4如何加上去

对三只箱子（羊）进行数据添加

datas[sx][sy] = 4;

继续箱子进行做碰撞检测

如果碰撞遇到箱子，想让箱子进行方向上的移动一格

箱子的数据移动

datas[wy][wx+1] = 0;

datas[wy][wx+1] = 4;

箱子的模型操作

要想让箱子移动，首先要获取箱子对应的对象JLabel

设计一个与数据模型完全相同的JLabel组件数组模型

JLabel[][] sheeps = new JLabel[12][16];

组件模型中当前是没有任何东西的，箱子在创建时必须要将其添加进去

sheeps[sx1][sy1] = lab\_sheep1;

sheeps[sx2][sy2] = lab\_sheep2;

sheeps[sx3][sy3] = lab\_sheep3;

人物的移动

参照之前，别忘了最后让操作结束，加一个return

总结：

人物碰撞实验要分三种情况

1.人物移动方向前方是空地

直接移动一位，修改人物的坐标

2.人物移动方向前方是树木

什么事情都不做

3.人物移动方向前方是箱子

3.1如果人物移动方向前方1位是箱子

人物移动一位，箱子移动一位

人物坐标修改，箱子对应坐标数据修改，箱子模型sheeps坐标修改

3.2如果人物移动方向前方1位是箱子，2位是箱子/树

什么事情都不做

当人物遇到目标（笼子）怎么办？

胜利判定

所谓胜利就是所有的箱子全部进入到目标中

设计一个变量，用于记录一共有多少个箱子

Int total = 3;

再设计一个变量，用于记录当前有多少个箱子进入到目标中

Int num = 0;

如果上面两个变量相等了，胜利，然后需要做什么事情，再设定；

什么时候改变num值？

当箱子进入到目标时，num+1

当箱子离开到目标时，num-1

当箱子从目标中进入到另一个目标中，num不变

考虑一下整体模型的设计

模型中1代表树木

树木 1

箱子 4

目标 8

箱子进入到目标 12

13种情况

------------------------------

人物 空地

人物 目标

人物显示位置改变 人物数据改变

-----------------------------

人物 树木

人物 箱子 树木

人物 箱子 箱子

人物 箱子 箱子目标

人物 箱子目标 树木

人物 箱子目标 箱子

人物 箱子目标 箱子目标

什么都不做

--------------------------------------------------------

人物 箱子 空地

人物显示位置改变 箱子显示位置改变 箱子模型改变 4-0 0-4

人物 箱子 目标

人物显示位置改变 箱子显示位置改变 箱子模型改变 4-8 8-12 num+1

人物 箱子目标 空地

人物显示位置改变 箱子显示位置改变 箱子模型改变 12-0 0-4 num-1

人物 箱子目标 目标

人物显示位置改变 箱子显示位置改变 箱子模型改变 12-8 8-12

箱子移动至目标结算

箱子移动出目标结算