目录

Day08.Java

1 final

- 三种用法
  - 常量
  - 方法
  - 类

1.1 常量

- 值不可变

```
final int a = 5;
a = 6;//错

final Dog d = new Dog("A", 50, 50);
d.name = "B";//对
d.full = 60;//对
d = new Dog(...);//错
d = null;//错
```

1.2 方法

方法不能被子类重写

1.3 类

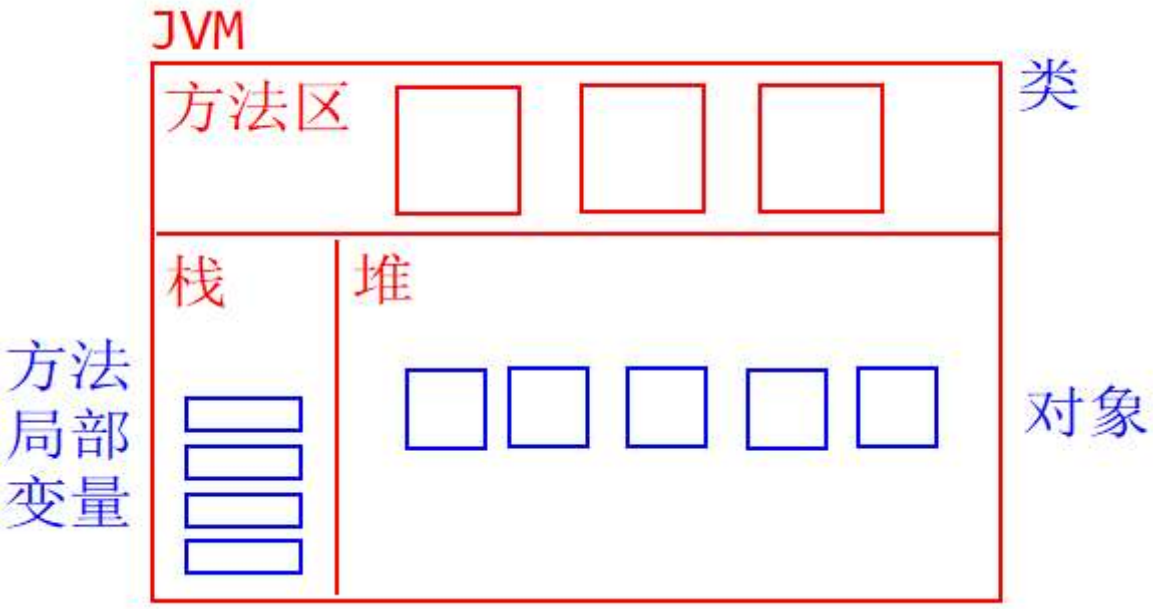
不能被继承，没有子类
System
String
Integer

2 static

- 静态
- 静态属于类，而不属于对象
- 静态变量，保存在方法区，在类的内存空间中保存

```
class Soldier {
   int id;
   static int count;//士兵数量
}
```

- 静态什么时候使用?
  - 使用原则： 能不用就不用
    静态是“非面向对象”的语法
  - 使用场景:
    - 共享的数据
    - 工具方法
      Math.random()
      Math.sqrt()
      Integer.parseInt()
      String.valueOf()

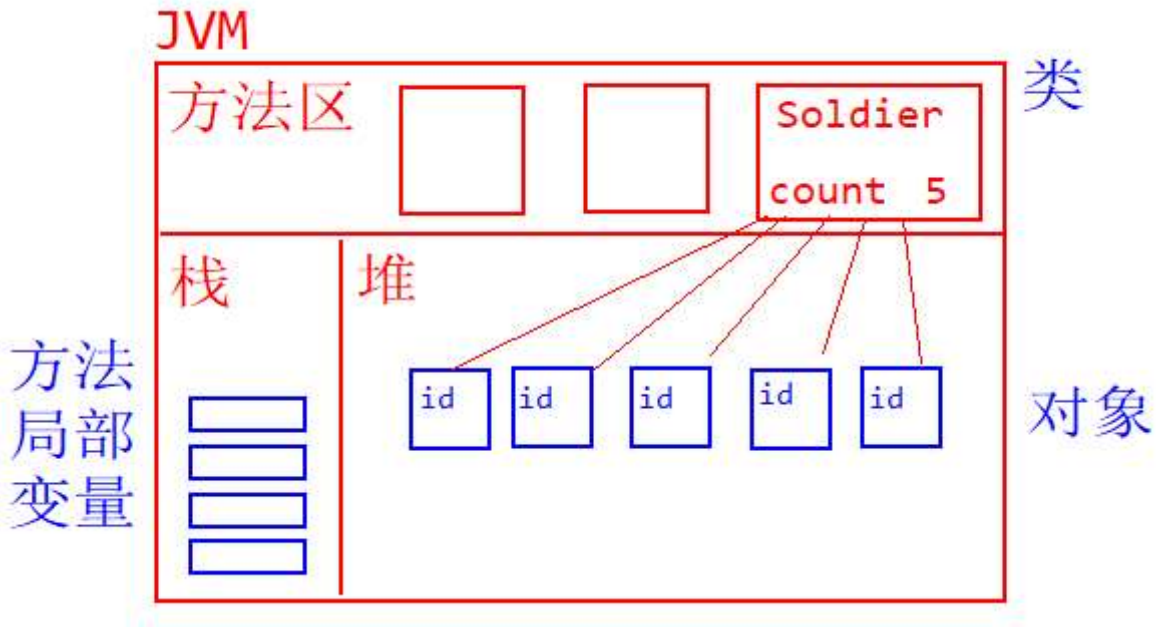


- 静态方法中，只能直接调用静态成员，而不能直接调用非静态成员
- 静态初始化块

```
class A {
   static {
      静态初始化块
      类被加载到"方法区"时，只执行一次
   }
}
```

静态

day0601_士兵
复制
day0801_士兵



Soldier

```
package day0801;

import java.util.Random;

/*
 * 封装
 * 把士兵相关的属性数据，和逻辑运算方法，
 * 封装成一个“类”组件
 */
public class Soldier {
   /*
    * 属性变量
    * 成员变量
    */
   int id; //默认值0
   int blood = 100;

   //静态士兵计数变量
   static int count;

   //构造方法
   public Soldier() {
      System.out.println("Soldier构造器");
      Soldier.count++;
   }

   /*
    * 成员方法
    */
   public void go() {
      System.out.println(id+"号士兵前进");
```

```
    }

    public void attack() {
        if(blood == 0) {
            System.out.println("这是"+id+"号士兵的尸体");
            return;
        }
        System.out.println(id+"号士兵进攻");
        //随机的减血量
        int d = new Random().nextInt(10);
        //减血
        blood -= d;
        if(blood<0) {
            blood = 0;
        }
        System.out.println("血量: "+blood);
        //血量是0, 阵亡
        if(blood == 0) {
            System.out.println(id+"号士兵阵亡");
            Soldier.count--;
        }
    }
}
```

Test2

```
package day0801;

import java.util.Scanner;

public class Test2 {
    public static void main(String[] args) {
        /*
         * 创建一组士兵对象,
         * 循环一轮一轮的进攻,
         * 直到所有士兵阵亡
         */
        //新建 Soldier[] 数组
        Soldier[] a = new Soldier[5];
        //遍历数组, 创建5个士兵对象, 存入数组
        for(int i=0;i<a.length;i++) {
            a[i] = new Soldier();
            a[i].id = i+1;
        }
        System.out.println(
         "已经创建"+Soldier.count+"个士兵");
        System.out.println("按回车进攻");
        //当还有存活的士兵
        while(Soldier.count != 0) {
            //遍历进攻
            new Scanner(System.in).nextLine();
            for(int i=0;i<a.length;i++) {
                a[i].attack();
```

```
            }
            System.out.println(
             "---------士兵数量："+Soldier.count);
        }


    }
}
```

3 常量

```
Integer.MAX_VALUE

public static final int MAX_VALUE=0x7fffffff;
```

- static final
  两个关键字定义常量
  - final 不可变
  - static 节省内存，只存一份

- 命名习惯：
  全大写，单词之间用下划线连接

4 访问控制符

- 控制一个类，或类中的成员的访问范围

	类	包	子类	任意
public				
protected				
[default]				
private				

- 选择的原则：
  - 尽量使用小范围
  - public 是与其他开发的一个契约，约定公开的东西，会尽量保持稳定不变

```
private
```

```
项目：day0802_学生
类：day0802.Test1
                Student
```

Student

```
package day0802;

public class Student {
   //成员变量，一般都设置成私有
   private int id;
   private String name;
   private String gender;
   private int age;

   public Student() {
      super();
   }
   public Student(int id, String name, String gender, int age) {
      super();
      this.id = id;
      this.name = name;
      this.gender = gender;
      this.age = age;
   }
   public int getId() {
      return id;
   }
   public void setId(int id) {
      this.id = id;
   }
   public String getName() {
      return name;
   }
   public void setName(String name) {
      this.name = name;
   }
   public String getGender() {
      return gender;
   }
   public void setGender(String gender) {
      this.gender = gender;
   }
   public int getAge() {
      return age;
   }
   public void setAge(int age) {
      this.age = age;
   }
}
```

Test

```
package day0802;
```

```
public class Test1 {
   public static void main(String[] args) {
      Student s = new Student();
      //s.id = 5;
      s.setId(5);
      s.setName("张三");
      s.setGender("男");
      s.setAge(23);

      //System.out.println(s.id);
      System.out.println(s.getId());
      System.out.println(s.getName());
      System.out.println(s.getGender());
      System.out.println(s.getAge());
   }
}
```

5 对象创建过程

```
class A {
   int v1 = 1;
   static int v2 = 2;
   static {}
   public A() {}
}

class B extends A {
   int v3 = 3;
   static int v4 = 4;
   static {}
   public B() {}
}


new B()
```

- 第一次用到A和B类
  1. 加载父类，为父类的静态变量分配内存
  2. 加载子类，为子类的静态变量分配内存
  3. 执行父类静态变量的赋值运算，和静态初始化块
  4. 执行子类静态变量的赋值运算，和静态初始化块
- 创建对象
  5. 创建父类对象，为父类的非静态变量分配内存
  6. 创建子类对象，为子类的非静态变量分配内存
  7. 父类的非静态变量赋值运算
  8. 执行父类构造方法
  9. 子类的非静态变量赋值运算

10 执行子类构造方法

对象创建过程

项目：day0803_对象创建过程
类：day0803.Test1

```
package day0803;

public class Test1 {
   public static void main(String[] args) {
      new B();
      System.out.println("-------------");
      new B();
   }
}

class A {
   int v1 = 1;
   static int v2 = 2;
   static {
      System.out.println("A静态块");
   }
   public A() {
      System.out.println("A构造方法");
   }
}
class B extends A {
   int v3 = 3;
   static int v4 = 4;
   static {
      System.out.println("B静态块");
   }
   public B() {
      System.out.println("B构造方法");
   }
}
```

飞机大战

项目：day0804_飞机大战
类：day0804.GamePanel
Main

图片压缩包中的 imgs 文件夹
鼠标拖拽到 eclipse 项目的 src 文件夹

```
*) BufferedImage 对象封装一张图片的数据
*) 加载图片 ImageIO.read(文件路径)
*) 文件路径使用一个工具，用图片的相对路径，来获取文件的绝对路径
    "/" 程序运行的目录，类和图片存放的目录 bin
    "/imgs/hero0.png"

    Main.class.getResource("/imgs/hero0.png")
```

Main

```
package day0804;

import java.awt.image.BufferedImage;

import javax.imageio.ImageIO;
import javax.swing.JFrame;

public class Main {
   static BufferedImage bg;
   static BufferedImage bullet;
   static BufferedImage start;
   static BufferedImage over;
   static BufferedImage pause;
   static BufferedImage[] airplane;
   static BufferedImage[] bigPlane;
   static BufferedImage[] bee;
   static BufferedImage[] hero;
   static {
      try {
                                                                  bg        =
ImageIO.read(Main.class.getResource("/imgs/background.png"));
                                                               bullet       =
ImageIO.read(Main.class.getResource("/imgs/bullet.png"));
                                                                start       =
ImageIO.read(Main.class.getResource("/imgs/start.png"));
                                                                 over       =
ImageIO.read(Main.class.getResource("/imgs/gameover.png"));
                                                               pause        =
ImageIO.read(Main.class.getResource("/imgs/pause.png"));
         airplane = new BufferedImage[5];
         for (int i = 0; i < airplane.length; i++) {
                                                           airplane[i]      =
ImageIO.read(Main.class.getResource("/imgs/airplane"+i+".png"));
         }

         bigPlane = new BufferedImage[5];
         for (int i = 0; i < bigPlane.length; i++) {
                                                           bigPlane[i]      =
ImageIO.read(Main.class.getResource("/imgs/bigplane"+i+".png"));
         }

         bee = new BufferedImage[5];
```

```
            for (int i = 0; i < bee.length; i++) {
                bee[i] = ImageIO.read(Main.class.getResource("/imgs/bee"+i+".png"));
            }

            hero = new BufferedImage[6];
            for (int i = 0; i < hero.length; i++) {
                hero[i] = ImageIO.read(Main.class.getResource("/imgs/hero"+i+".png"));
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        JFrame f = new JFrame();
        f.setTitle("飞机大战");
        f.setDefaultCloseOperation(
          JFrame.EXIT_ON_CLOSE);
        f.setResizable(false);
        //新建 GamePanel 对象，添加到窗口中
        GamePanel game = new GamePanel();
        f.add(game);
        //让窗口，恰好包裹内部的面板
        f.pack();

        //显示
        f.setVisible(true);
    }
}
```

GamePanel

```
package day0804;

import java.awt.Dimension;
import java.awt.Graphics;

import javax.swing.JPanel;

public class GamePanel extends JPanel {

    public GamePanel() {
        //设置面板的期望大小
        setPreferredSize(new Dimension(400, 654));
    }

    /*
     * 固定的绘图方法
     * 由系统自动调用
```

```
     */
    @Override
    public void paint(Graphics g) {
        g.drawImage(Main.bg,0,0,null);
        g.drawImage(Main.hero[0],180,400,null);
        g.drawImage(Main.bee[0],100,150,null);
    }
}
```

飞机大战

day0804_飞机大战
复制
day0805_飞机大战

背景 Background 类
属性:
    img = Main.bg
    x1
    y1
    x2
    y2
方法:
    paint(g) 用来再画布上，绘制自身
    step() 移动一步

Background

```
package day0804;

import java.awt.Graphics;
import java.awt.image.BufferedImage;

public class Background {
    BufferedImage img = Main.bg;
    int x1 =0;
    int y1 =0;
    int x2 =0;
    int y2 =-852;

    public void paint(Graphics g) {
        g.drawImage(img,x1,y1,null);
        g.drawImage(img,x2,y2,null);
```

```
    }

    public void step() {
        y1++;
        y2++;
        if(y1 == 852) {
            y1 = -852;
        }
        if(y2 == 852) {
            y2 = -852;
        }
    }
}
```

GamePanel

```
package day0804;

import java.awt.Dimension;
import java.awt.Graphics;

import javax.swing.JPanel;

public class GamePanel extends JPanel {
    Background bg = new Background();

    public GamePanel() {
        //设置面板的期望大小
        setPreferredSize(new Dimension(400, 654));
    }

    /*
     * 固定的绘图方法
     * 由系统自动调用
     */
    @Override
    public void paint(Graphics g) {
        bg.paint(g);
    }

    //动起来方法
    public void action() {
        //画面一帧一帧的循环播放
        while(true) {
            bg.step();//背景移动

            //通知底层系统，重绘画面
            //系统受到通知后，会自动调用 paint() 方法
            repaint();
            //暂停 1/60 秒 (60 fps)
            try {
                Thread.sleep(1000/60);
```

```
            } catch (InterruptedException e) {
            }
        }
    }
}
```

Main

```
package day0804;

import java.awt.image.BufferedImage;

import javax.imageio.ImageIO;
import javax.swing.JFrame;

public class Main {
    static BufferedImage bg;
    static BufferedImage bullet;
    static BufferedImage start;
    static BufferedImage over;
    static BufferedImage pause;
    static BufferedImage[] airplane;
    static BufferedImage[] bigPlane;
    static BufferedImage[] bee;
    static BufferedImage[] hero;
    static {
        try {
                                                    bg        =
ImageIO.read(Main.class.getResource("/imgs/background.png"));
                                                  bullet      =
ImageIO.read(Main.class.getResource("/imgs/bullet.png"));
                                                   start      =
ImageIO.read(Main.class.getResource("/imgs/start.png"));
                                                    over      =
ImageIO.read(Main.class.getResource("/imgs/gameover.png"));
                                                   pause      =
ImageIO.read(Main.class.getResource("/imgs/pause.png"));
            airplane = new BufferedImage[5];
            for (int i = 0; i < airplane.length; i++) {
                                                 airplane[i]   =
ImageIO.read(Main.class.getResource("/imgs/airplane"+i+".png"));
            }

            bigPlane = new BufferedImage[5];
            for (int i = 0; i < bigPlane.length; i++) {
                                                 bigPlane[i]   =
ImageIO.read(Main.class.getResource("/imgs/bigplane"+i+".png"));
            }

            bee = new BufferedImage[5];
            for (int i = 0; i < bee.length; i++) {
                                                    bee[i]     =
ImageIO.read(Main.class.getResource("/imgs/bee"+i+".png"));
            }
```

```
            hero = new BufferedImage[6];
            for (int i = 0; i < hero.length; i++) {
                                                                hero[i]      =
ImageIO.read(Main.class.getResource("/imgs/hero"+i+".png"));
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        JFrame f = new JFrame();
        f.setTitle("飞机大战");
        f.setDefaultCloseOperation(
          JFrame.EXIT_ON_CLOSE);
        f.setResizable(false);
        //新建 GamePanel 对象，添加到窗口中
        GamePanel game = new GamePanel();
        f.add(game);
        //让窗口，恰好包裹内部的面板
        f.pack();

        //显示
        f.setVisible(true);

        //动起来
        game.action();
    }
}
```

飞机大战

day0805_飞机大战
复制
day0806_飞机大战

Enemy
    |- Airplane
    |- BigPlane
    |- Bee

Enemy
属性:
    imgs 一组图片
    img 当前使用的图片
    x
    y

方法:
paint(g)
step() 抽象方法
rndX() 随机定位水平的出现位置

Enemy

```
package day0804;

import java.awt.Graphics;
import java.awt.image.BufferedImage;
import java.util.Random;

public abstract class Enemy {
    //初始值，要在子类构造方法中赋值
    BufferedImage[] imgs;
    BufferedImage img;
    int x;
    int y;

    public void paint(Graphics g) {
        g.drawImage(img,x,y,null);
    }

    public abstract void step();

    public int rndX() {
        // [0, 400-图片宽度)
        return new Random().nextInt(400-img.getWidth());
    }
}
```

Airplane

```
package day0804;

public class Airplane extends Enemy {

    public Airplane() {
        imgs = Main.airplane;
        img = imgs[0];
        x = rndX();
        y = -img.getHeight();
    }

    @Override
    public void step() {
        y += 4;
    }
```

```
}
```

BigPlane

```
package day0804;

public class BigPlane extends Enemy {

    public BigPlane() {
        imgs = Main.bigPlane;
        img = imgs[0];
        x = rndX();
        y = -img.getHeight();
    }

    @Override
    public void step() {
        y += 2;
    }

}
```

Bee

```
package day0804;

public class Bee extends Enemy {
    int dx;

    public Bee() {
        imgs = Main.bee;
        img = imgs[0];
        x = rndX();
        y = -img.getHeight();
        dx = (Math.random()<0.5 ? -2 : 2);
    }

    @Override
    public void step() {
        y += 3;
        x += dx;
    }

}
```

GamePanel

```
package day0804;

import java.awt.Dimension;
```

```
import java.awt.Graphics;

import javax.swing.JPanel;

public class GamePanel extends JPanel {
    Background bg = new Background();
    Enemy[] ememys = {
            new Airplane(),
            new BigPlane(),
            new Bee()
    };

    public GamePanel() {
        //设置面板的期望大小
        setPreferredSize(new Dimension(400, 654));
    }

    /*
     * 固定的绘图方法
     * 由系统自动调用
     */
    @Override
    public void paint(Graphics g) {
        bg.paint(g);
        for (int i = 0; i < ememys.length; i++) {
            Enemy e = ememys[i];
            e.paint(g);
        }
    }

    //动起来方法
    public void action() {
        //画面一帧一帧的循环播放
        while(true) {
            bg.step();//背景移动
            for (int i = 0; i < ememys.length; i++) {
                Enemy e = ememys[i];
                e.step();
            }

            //通知底层系统，重绘画面
            //系统受到通知后，会自动调用 paint() 方法
            repaint();
            //暂停 1/60 秒 (60 fps)
            try {
                Thread.sleep(1000/60);
            } catch (InterruptedException e) {
            }
        }
    }
}
```

6 作业

- 重写
  - 0804不用再写
  - day0805_飞机大战
    - 背景
    - Background
    - GamePanel 显示背景，移动
    - Main 调用动起来方法
  - day0806_飞机大战
    - 敌人
    - Enemy,Airplane,BigPlane,Bee
    - GamePanel 添加敌人，绘制、移动
- 复习面向对象概念