

实验内容	第 12 周 Java 小应用程序			成 绩	
姓 名	王秋锋	学号	2015111948	班 级	计算机 2015-03 班
专 业	计算机科学与技术			日 期	2017 年 11 月 24 日

## 【实验目的】--字符串处理

- ◆ 掌握 Java 小应用程序；
- ◆ 掌握 JAVA GUI 中文本输出，图形绘制，图像输出
- ◆ 第一个课程设计发布

## 【实验内容】

### 1、编辑、编译、运行下面 java 程序（从 html 中传递参数）

理解掌握小应用程序的运行过程，Font 类的使用

```
import java.awt.*;
import java.applet.*;
public class AppletPara extends Applet
{
    String s1,s2;
    Font msgFont=new Font("TimesRoman",Font.ITALIC,20); //生成新字体
    public void init()
    {
        s1 = getParameter("p1"); //从 html 页面中获取参数值
        s2 = getParameter("p2");
    }
    public void paint(Graphics g)
    {
        g.setFont(msgFont); //设置字体
        g.setColor(Color.blue); //设置演示
        g.drawString("我来自：",10,40); 绘制文本
        g.drawString(s1,10,80);
        g.drawString(s2,10,120);
    }
}
```

\*\*\*\*\*下面是嵌入小应用程序的 html 文件

```
<HTML>
<HEAD>
<TITLE>Applet Parameter Test</TITLE>
</HEAD>
<applet code= "AppletPara.class" width=300 height=300>
<param name=p1 value= "西南交大">
<param name=p2 value= "信息学院">
</applet>
</HTML>
```

要求：

(1) 分析该程序，写出运行结果

### 【实验结果与分析】

实验分析：

Java 的 font 类

设计字体显示效果 `Font mf = new Font(String 字体, int 风格, int 字号);`

字体: TimesRoman, Courier, Arial 等


风格: 三个常量 `IFont.PLAIN`, `Font.BOLD`, `Font.ITALIC`

字号: 字的大小 (磅数)

设置组件当前使用的字体: `setFont(Font fn)`

获取组件当前使用的字体: `getFont()`

实验结果：

 小应用程序查看器: AppletPara.class      —      □      ×

小应用程序

我来自:

西南交大

信息学院

## 2. 编辑、编译、运行下面 java 程序

理解掌握 `paint` 方法, `grapgics` 类, `Color` 类的应用

```
import java.awt.Graphics;
import java.awt.Color;
import java.awt.Font;
public class Demo extends java.applet.Applet {
    public void paint(Graphics g) {
        g.drawLine(6,150,50,50);    //绘制直线
        g.drawLine(6,150,50,150);
        g.drawRect(60,30,100,30);    //绘制矩形
        g.draw3DRect(60,160,100,30,true); //绘制 3D 矩形，true 表示 3D 效果是凸的
        g.drawRoundRect(60,200,100,40,10,10);    //绘制圆角矩形
        g.drawArc(200,50,10,8,0,45);
        g.drawOval(200,150,80,80);
        g.drawString("测试绘制文字",200,250);    //绘制字符串
        Color c = Color.RED;    //创建一个红色颜色对象
        g.setColor(c);    //设置绘图颜色为红色
        g.fillOval(200,100,100,40);    //绘制填充椭圆
        g.setColor(Color.PINK);
        g.fill3DRect(60,80,100,30,true);    //绘制三维矩形
        g.setColor(Color.BLUE);
        g.fillArc(200,200,100,80,0,45);    //绘制填充弧
        Font f = new Font("黑体",Font.ITALIC,60);    //创建字体对象
        g.setFont(f);    //设置绘图字体
        g.drawString("欢迎访问",10,300);    //绘制字符串
    }
}
```

要求：分析该程序，给出运行结果

### 【实验结果与分析】

实验分析：

Java 提供的 Graphics 类是用于绘图和显示格式化文字的工具。绘图必须在一个窗口（容器）中进行，Java 小程序 Applet 就是一个窗口。

在 Applet 中写字和画图就要创建一个 Applet，必须让自己的类从 java.applet.Applet 类继承即可。Applet 类是一个抽象类，它提供了五个抽象方法，其中 paint() 是在 applet 窗口中或画布上写字画图。

Graphics 类在 java.awt 包中被声明。AWT(Abstract Windows Toolkit)的名称是抽象窗口工具包，是提供窗口及其组件的类库。写字和画图是用 Graphics 的 drawXXX 方法实现的。如 drawString(String),drawLine(.)等。画图用的坐标系是原点在左上角，纵轴向下以像素为单位的坐标系。

设置背景色的是方法继承自 Applet 的，语法为：

```
setBackground(new Color(int,int,int))
```

即它的参数是一个 Color 对象句柄。

设置前景色的方法是属于 Graphics 的，即设置 Graphics 的绘图色。语法为：

```
g.setColor(Color 对象);
```


选择颜色有两种方法，一是直接用颜色值 RGB 创建 Color 对象，如：

```
Color color = new Color(R,G,B)
```

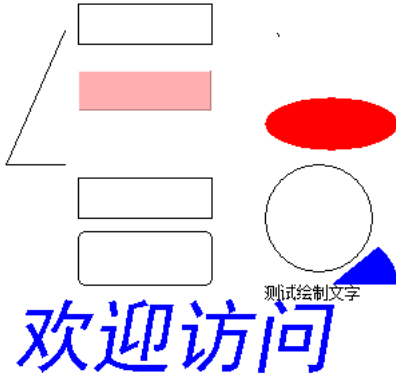
另一种是用颜色常量如 Color.red, Color.green 等。

本程序就使用了 graphics 类，Color 类绘制了各种图形。

实验结果：

 小应用程序查看器: aa.Demo.class

小应用程序



### 3、编辑并运行下面程序，理解掌握 applet 载入图片

```
import java.awt.*;
import java.applet.*;
public class UseImage extends Applet    //定义图像对象
{
    Image testImage;
    public void init()                  //得到图片
    {
        testImage = getImage(getDocumentBase(),"flag.Gif"); //图片与 html 文件在一个子目录中
    }
    public void paint(Graphics g)
    {
        g.drawImage(testImage,0,0,this);
    }
}
```

要求：分析该程序，给出运行结果

#### 【实验结果与分析】

实验分析：

getDocumentBase()可以获取文件的基本路径

Graphics 的 drawImage 方法有 6 个，这里用的是 public abstract boolean drawImage(Image img, int x, int y, ImageObserver observer);

img 是要加载的图像，x，y 是指定绘制图像矩形左上角的位置，observer 是要绘制图像的容器。

实验结果：



#### 4、编辑并运行下面程序，理解掌握在 applet 中布局 GUI

```
import java.awt.*;
import java.applet.*;

public class TestAppletGui extends Applet
{
    Label lblUserName=new Label("用户姓名: ");
    TextField txtUserName=new TextField (12);

    Label lblUserPasswd=new Label("用户密码: ");
    TextField txtUserPasswd=new TextField (12);

    Button btnLogin=new Button("登录");
    Button btnReset =new Button("重置");

    public void init()
    {
        //设置密码的回显字符为*
        txtUserPasswd.setEchoChar('*');

        add(lblUserName);
        add(txtUserName);

        add(lblUserPasswd);
        add(txtUserPasswd);

        add(btnLogin);
        add(btnReset);
    }
}
```

要求：分析该程序，给出运行结果

#### 【实验结果与分析】

实验分析：

本程序涉及到的 GUI 有：

Button——按钮

Label——标签

TextField——文本区

setEchoChar 将密码变成提示字符,比如输密码的时候,屏幕不可能让别人看见输入的密码是什么,但是又必须知道在输密码,且输了几位.就显示这个符号.比如设置\*,输密码的时候显示的就是\*

实验结果：



## 5、编辑并运行下面程序，理解掌握画布的使用

```
import java.awt.*;
import java.awt.event.*;
class MyCanvas extends Canvas{ //定义画布
    int x,y,r;
    MyCanvas()
    {setBackground(Color.blue);}
    public void setX(int x)
    { this.x=x;}
    public void setY(int y)
    {this.y=y;}
    public void setR(int r) //设置半径
```

```
{ this.r=r;}

public void paint(Graphics g)  //绘制圆
{ setForeground(Color.red);
  g.drawOval(x, y, 2*r, 2*r);
}
}

class WindowCavans extends Frame implements ActionListener
{
    MyCanvas canvas;    TextField inputR, inputX, inputY;    Button btn;
    WindowCavans(){
        canvas=new MyCanvas();    //画布对象
        inputR=new TextField("40",5);
        inputX=new TextField("100",4);
        inputY=new TextField("30",4);
        Panel pNorth=new Panel(), pSouth=new Panel();
        pNorth.add(new Label("圆的位置坐标: "));
        pNorth.add(inputX);
        pNorth.add(inputY);
        pSouth.add(new Label("圆的半径: "));
        pSouth.add(inputR);
        btn=new Button("确定");
        btn.addActionListener(this);
        pSouth.add(btn);
        this.add(pNorth, BorderLayout.NORTH);
        this.add(pSouth, BorderLayout.SOUTH);
        this.add(canvas, BorderLayout.CENTER);    //画布放置在中心区域
        setBounds(100, 100, 500, 500);    //指定窗口的位置与大小
        setVisible(true);    }
        public void actionPerformed(ActionEvent e)
        {    int x,y,r;
            try
            {
                x=Integer.parseInt(inputX.getText());
                y=Integer.parseInt(inputY.getText());
                r=Integer.parseInt(inputR.getText());
                canvas.setX(x);
                canvas.setY(y);
                canvas.setR(r);
                canvas.repaint();
            }
            catch(Exception ex)
            {x=0;y=0;r=0;}
        }
    }
}
```

```
public class TestCanvas
{
    public static void main(String[] args)
    {new WindowCavans();}
}
```

**要求：**（1）运行程序，给出正确的程序运行结果，分析程序的功能。

### 【实验结果与分析】

实验分析：

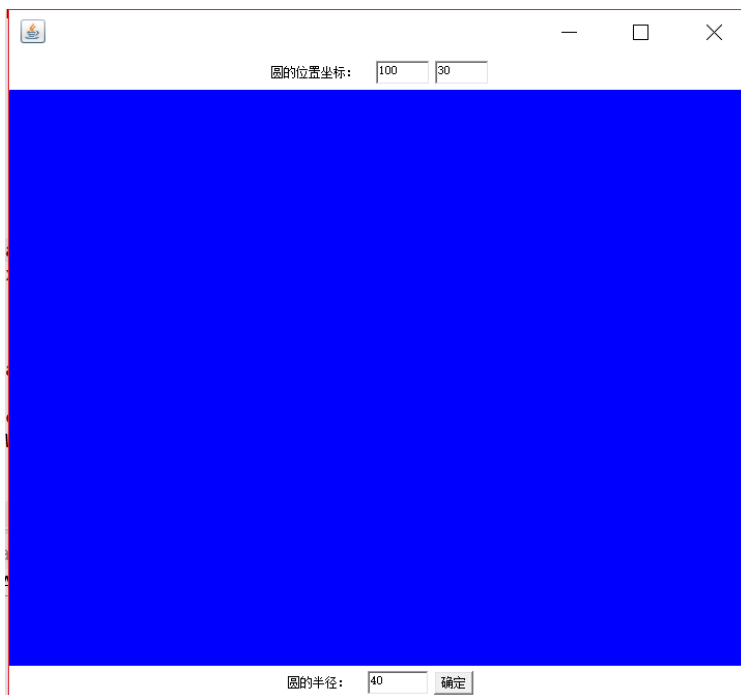
Canvas 组件表示屏幕上一个空白矩形区域，应用程序可以在该区域内绘图，或者可以从该区域捕获用户的输入事件。

应用程序必须为 Canvas 类创建子类，以获得有用的功能（如创建自定义组件）。必须重写 paint 方法，以便在 canvas 上执行自定义图形。

本程序用 Canvas 组件创建了蓝色的矩形画图区域，重写了 paint 方法，每次调用会绘制出红色的圆。

除此之外还布局了 GUI，用户可以输入圆心坐标和半径然后绘制出圆，本程序用了 ActionListener 里的 actionPerformed 来接受用户输入的参数。

实验结果：



初始的时候圆心的坐标为(100,30),半径为 40 按下确定之后，圆就绘制出来了，然后再绘制圆心在(200,100),半径为 100 的圆



