第八章 绘图及AWT图形化用户界面

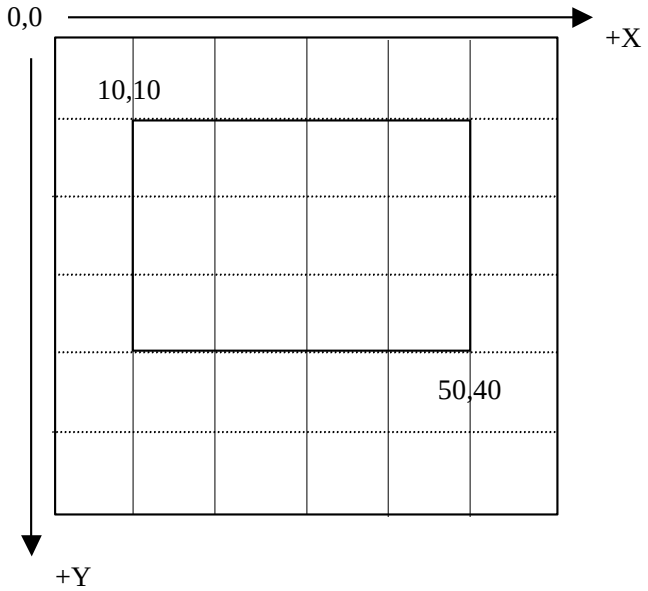
- §8.1绘图
- §8.2AWT图形化用户界面

8.1 绘图

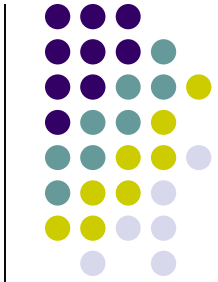
- 8.1.1 图形环境和图形对象
- 8.1.2 颜色和字体
- 8.1.3 使用Graphics类绘图
- 8.1.4 使用Graphics2D类绘图

8.1.1 图形环境和图形对象

- Java的图形坐标系统
  - 为了将某一图形在屏幕上绘制出来，首先要确定图形的位置，为了解决这个问题就必须有一个精确的图形坐标系统来定位图形。
  - GUI组件的左上角坐标默认为(0, 0)
  - 从左上角到右下角，水平坐标x和垂直坐标y增加。
  - 坐标的单位是象素，所有坐标点的值都取整数。



8.1.1 图形环境和图形对象



- **Graphics对象**
  - **Graphics对象是专门管理图形环境的。Graphics类是一个抽象类**
  - **设计一个抽象类Graphics可以给程序员提供一个与平台无关的绘图接口，因而程序员就可以以独立于平台的方式来使用图形。**
  - **在各个平台上实现的Java系统将创建Graphics类的一个子类，来实现绘图功能，但是这个子类对程序员是透明的，也就是说我们只能看得到Graphics类，却不必关心其实现。**
  - **在执行paint方法时，系统会传递一个指向特定平台的Graphics子类的图形对象g**

8.1.2 颜色和字体

- Java中有关颜色的类是Color类，它在java.awt包中，这个类声明了用于操作Java程序中颜色的方法和常量。

名称	描述
public final static Color GREEN	常量 绿色
public final static Color RED	常量 红色
public Color(int r,int g,int b)	通过指定红、蓝、绿颜色分量（0～255），创建一种颜色
public int getRed()	返回某颜色对象的红色分量值(0～255)
Graphics: public void setColor(Color c)	Graphics类的方法，用于设置组件的颜色
Graphics: public Color getColor()	Graphics类的方法，用于获得组件的颜色

8.1.2 颜色和字体

- Font类——有关字体控制，在java.awt包中

名称	描述
public final static int PLAIN	一个代表普通字体风格的常量
public final static int BOLD	一个代表黑体字体风格的常量
public final static int ITALIC	一个代表斜体字体风格的常量
public Font(String name,int style,int size)	利用指定的字体、风格和大小创建一个Font对象
public int getStyle()	返回一个表示当前字体风格的整数值
public Boolean isPlain()	字体是否是普通字体风格
Graphics：public Font getFont()	获得当前字体
Graphics：public void setFont(Font f)	设置当前字体为f指定的字体、风格和大小

8.1.3 使用Graphics类绘图

- Graphics类
  - 其对象可以绘制文本、线条、矩形、多边形、椭圆、弧等多种图形

名称	描述
public void drawString(String str, int x, int y)	绘制字符串，左上角的坐标是（x,y)
public void drawLine(int x1, int y1, int x2, int y2)	在(x1,y1)与(x2,y2)两点之间绘制一条线段
public void drawRect(int x, int y, int width, int height)	用指定的width和height绘制一个矩形，该矩形的左上角坐标为(x, y)
public void fillRect(int x, int y, int width, int height)	用指定的width和height绘制一个实心矩形，该矩形的左上角坐标为(x,y)

8.1.3 使用Graphics类绘图

名称	描述
public void clearRect(int x, int y, int width, int height)	用指定的width和height，以当前背景色绘制一个实心矩形。该矩形的左上角坐标为（x,y)
public void drawRoundRect(int x, int y, int width, int height, int arcWidth, int arcHeight)	用指定的width和height绘制一个圆角矩形，圆角是一个椭圆的1/4弧，此椭圆由arcWidth、arcHeight确定两轴长。其外切矩形左上角坐标为（x,y)
public void fillRoundRect(int x, int y, int width, int height, int arcWidth, int arcHeight)	用当前色绘制实心圆角矩形，各参数含义同drawRoundRect。
public void draw3DRect(int x, int y, int width, int height, boolean b)	用指定的width和height绘制三维矩形，该矩形左上角坐标是(x,y)，b为true时，该矩形为突出的，b为false时，该矩形为凹陷的。
public void fill3DRect(int x, int y, int width, int height, boolean b)	用当前色绘制实心三维矩形，各参数含义同draw3DRect。

8.1.3 使用Graphics类绘图

名称	描述
public void drawPolygon(int[] xPoints, int [] yPoints, int nPoints)	用xPoints，yPoints数组指定的点的坐标依次相连绘制多边形，共选用前nPoints个点。
public void fillPolygon(int[] xPoints, int [] yPoints, int nPoints)	绘制实心多边形，各参数含义同drawPolygon。
public void drawOval(int x, int y, int width, int height)	用指定的width和height，以当前色绘制一个椭圆，外切矩形的左上角坐标是(x,y)。
public void fillOval(int x, int y, int width, int height)	绘制实心椭圆，各参数含义同drawOval。
public void drawArc(int x, int y,int width, int height, int startAngle, int arcAngle)	绘制指定width和height的椭圆，外切矩形左上角坐标是(x,y)，但只截取从startAngle开始，并扫过arcAngle度数的弧线。
public void fillArc(int x, int y,int width, int height, int startAngle, int arcAngle)	绘制一条实心弧线（即扇形），各参数含义同drawArc

8.1.3 使用Graphics类绘图

- 用各种颜色绘制文字及各种图形

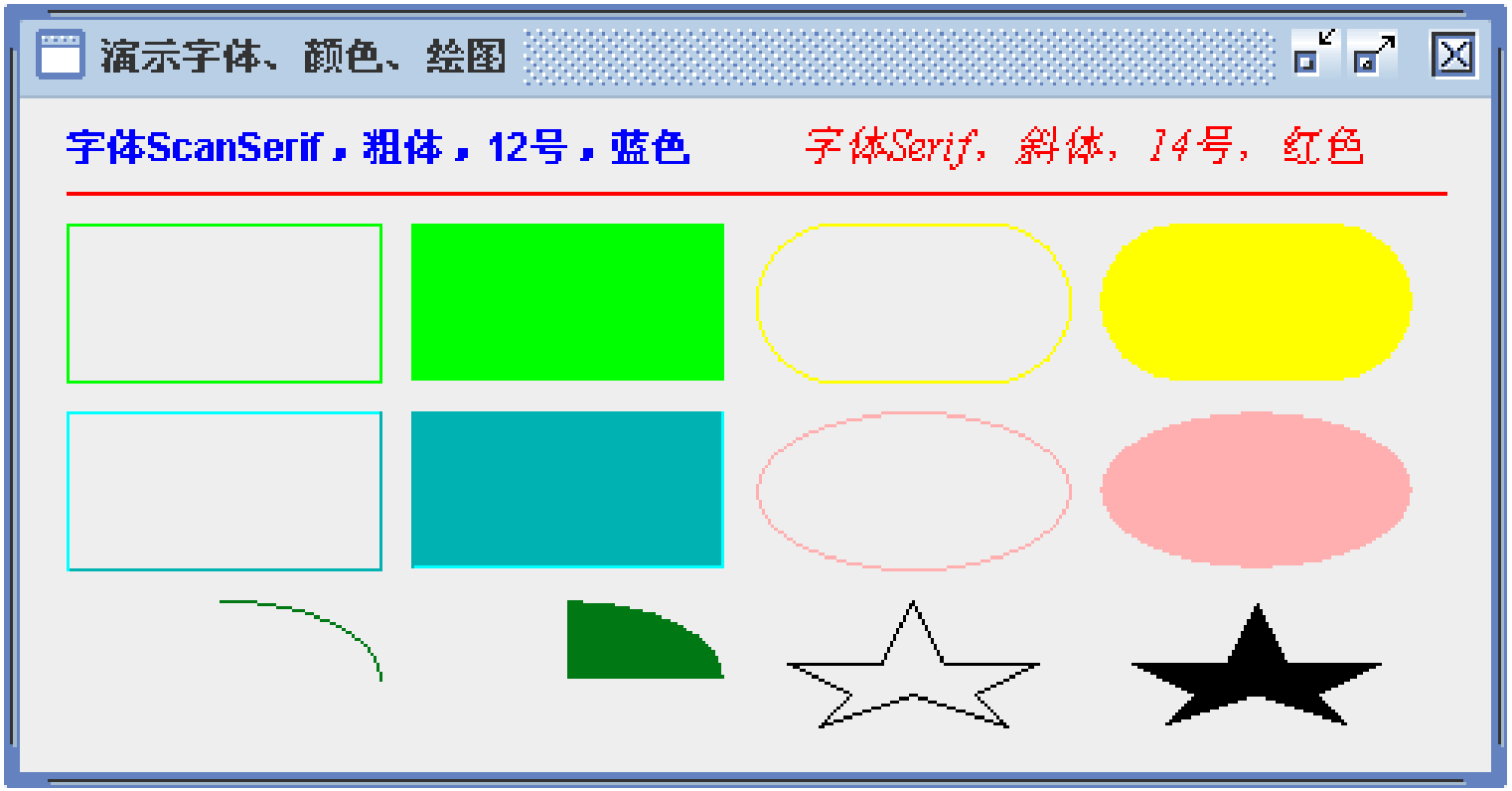
```
import java.awt.*;
import javax.swing.*;
public class Ex8_1 extends JFrame {
    public Ex81() {
      super( “演示字体、颜色、绘图” );    //调用基类构造方法
      setSize( 480, 250 );                //设置窗口大小
      setVisible( true );                 //显示窗口
   }
   public void paint( Graphics g )  {
       super.paint( g );  // call superclass's paint method
       g.setFont( new Font( "宋体", Font.BOLD, 12 ) );
       g.setColor(Color.blue);                          //设置颜色
       g.drawString("字体ScanSerif, 粗体, 12号, 蓝色", 20, 50);
```

8.1.3 使用Graphics类绘图

```
g.setFont( new Font( "Serif“, Font.ITALIC, 14 ) );
g.setColor(new Color(255,0,0));
g.drawString( " 字体Serif, 斜体, 14号, 红色",  250,  50 );

g.drawLine(20, 60, 460, 60);                    //绘制直线

g.setColor(Color.green);
g.drawRect(20, 70, 100, 50);                    //绘制空心矩形
g.fillRect(130, 70, 100, 50);                   //绘制实心矩形

g.setColor(Color.yellow);
g.drawRoundRect(240, 70, 100, 50, 50, 50); //绘制空心圆角矩形
g.fillRoundRect(350, 70, 100, 50, 50, 50);      //绘制实心圆角矩形

g.setColor(Color.cyan);
g.draw3DRect(20, 130, 100, 50, true);   //绘制突起效果空心矩形
g.fill3DRect(130, 130, 100, 50, false);    //绘制凹陷效果实心矩形
```

```
    g.setColor(Color.pink);
    g.drawOval(240,130,100,50);              //绘制空心椭圆
    g.fillOval(350,130,100,50);              //绘制实心椭圆
    g.setColor(new Color(0,120,20));
    g.drawArc(20,190,100,50,0,90);           //绘制一段圆弧
    g.fillArc(130,190,100,50,0,90);          //绘制扇形

    g.setColor(Color.black);
    int xValues[]={250,280,290,300,330,310,320,290,260,270};
    int yValues[]={210,210,190,210,210,220,230,220,230,220};
    g.drawPolygon(xValues,yValues,10);      //绘制空心多边形

    int xValues2[]={360,390,400,410,440,420,430,400,370,380};
    g.fillPolygon(xValues2,yValues,10);        //绘制实心多边形
  }
  public static void main( String args[] )  {
    JFrame.setDefaultLookAndFeelDecorated(true);
    //设置窗口的外观感觉为Java默认
    Ex8_1 application = new Ex8_1();
     application.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE );
  }
}
```

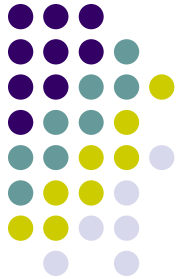
- 运行结果



8.1.4 使用Graphics2D类绘图

- **Java2D API**
  - **提供了高级的二维图形功能**
  - **分布在java.awt、java.awt.image、java.awt.color、java.awt.font、java.awt.geom、java.awt.print和java.awt.image.renderable包中**
  - **它能轻松使你完成以下功能：**
    - **绘制任何宽度的直线**
    - **用渐变颜色和纹理来填充图形**
    - **平移、旋转、伸缩、切变二维图形，对图像进行模糊、锐化等操作**
    - **构建重叠的文本和图形**

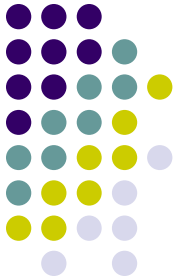
8.1.4 使用Graphics2D类绘图-续



- **Graphics2D类**
  - **要想使用Java2D API，就必须通过一个该类的对象是Graphics类的抽象子类**
  - **事实上，所有的paint方法用于绘图操作的对象实际上是Graphics2D的一个子类实例，该实例传递给paint方法，并被向上转型为Graphics类的实例。**
  - **要访问Graphics2D功能，必须使用如下语句将传递给paint方法的Graphics引用强制转换为Graphics2D引用：**

    Graphics2D g2d = (Graphics2D)g

8.1.4 使用Graphics2D类绘图



- 使用Java2D使文字出现渐变色效果

```
import java.awt.*;
import javax.swing.*;
public class Ex8_2 extends JApplet{
   public void paint(Graphics g) {
     super.paint(g);
     Graphics2D g2d=(Graphics2D)g;
     g2d.setPaint(new GradientPaint(0,0,Color.red,180,45,Color.yellow));
     g2d.drawString("This is a Java Applet!", 25, 25);
   }
}
```

8.1.4 使用Graphics2D类绘图

- 编译ex8_2.java产生字节码文件ex8_2.class。接下来就需要编写一个HTML文件ex8_2.html来嵌入ex8_2.class

```
<html>
<applet code="Ex8_2.class" width="300" height="45">
</applet>
</html>
```

- 将ex8_2.html文件和Ex8_2.class文件放在同一个目录下。现在，在浏览器中打开这个HTML文件，当浏览器遇到Applet标记时，就会自动载入指定的class文件，就会实现在屏幕上绘制一串字符的效果

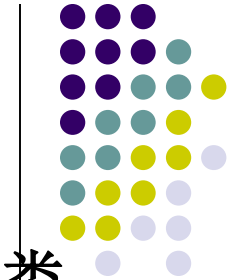
8.2 AWT图形化用户界面

- 前面介绍了如何在屏幕上绘制普通的图形，但如果需要绘制一个按钮，并使其可以对点击事件作出响应，就需要使用**java.AWT, javax.Swing**提供的组件
- **JFC**
  - **Java Foundation Classes（Java基础类）的缩写**
  - **是关于GUI 组件和服务的完整集合**
  - **作为J2SE 的一个有机部分，主要包含5 个部分**
    - **AWT**
    - **Java2D**
    - **Accessibility**
    - **Drag & Drop**
    - **Swing**

8.2 AWT图形化用户界面

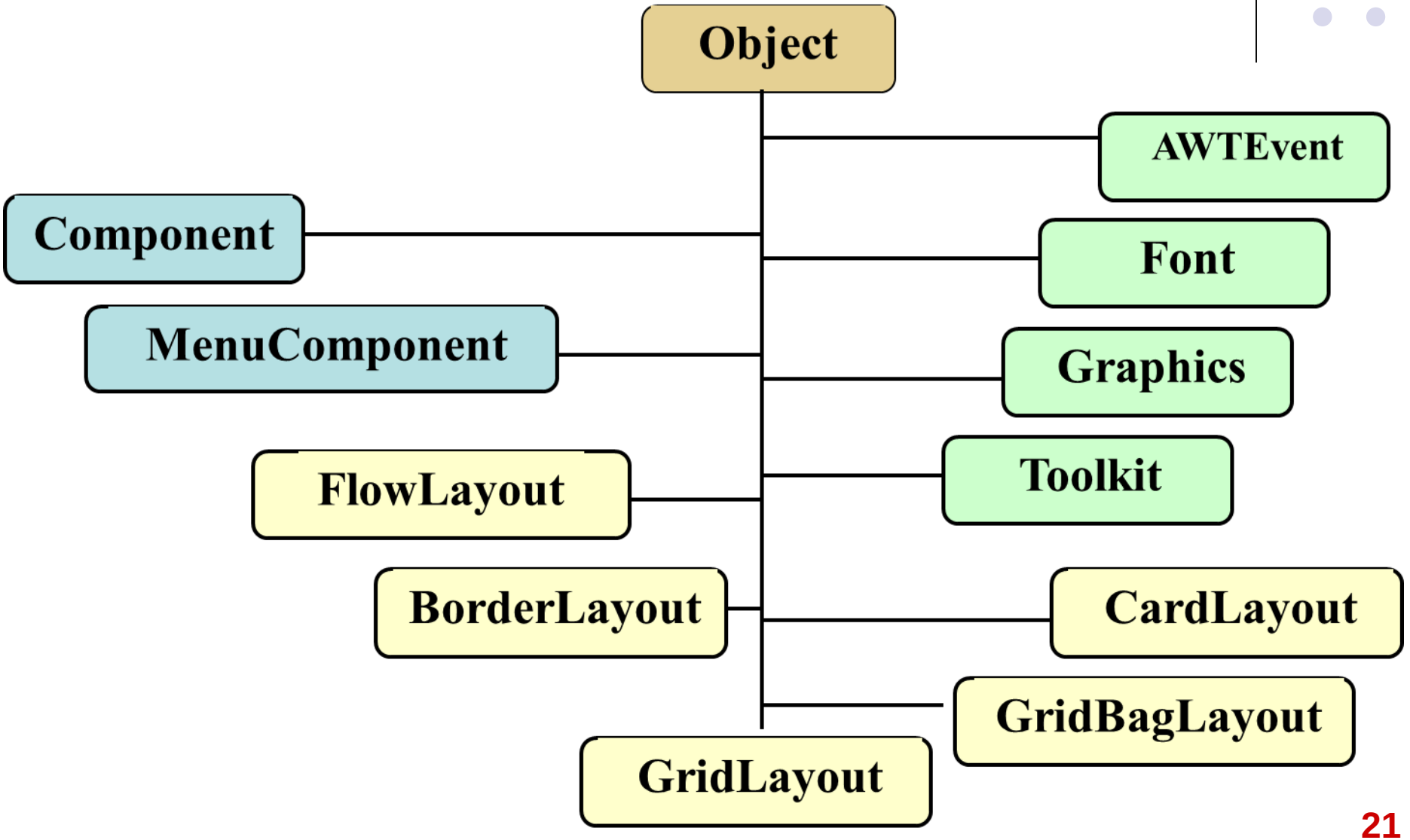
- 开发图形界面的应用程序时，需要用到AWT(抽象图形工具集)。AWT是Java开发工具包(JDK)的一部分，是Java基本类(JFC)的核心。
- AWT的作用是给用户提供基本的界面组件，如：窗口、按键、菜单等。
- javax.swing包是Java2新增的图形界面类库。Swing是基于AWT基本结构创建的二级用户界面工具集。与旧的AWT相比，Swing提供更加丰富的组件集，Swing中所提供的组件集几乎可以替代所有AWT中原有的组件。由此，许多人也许会产生Swing是AWT的替代物的误解，而实际上Swing是基于AWT之上创建的。因此，为了准确的掌握Swing组件集的工作方式，必须首先掌握AWT组件的行为及其工作原理。

8.2.1 java.awt包及类的层次结构

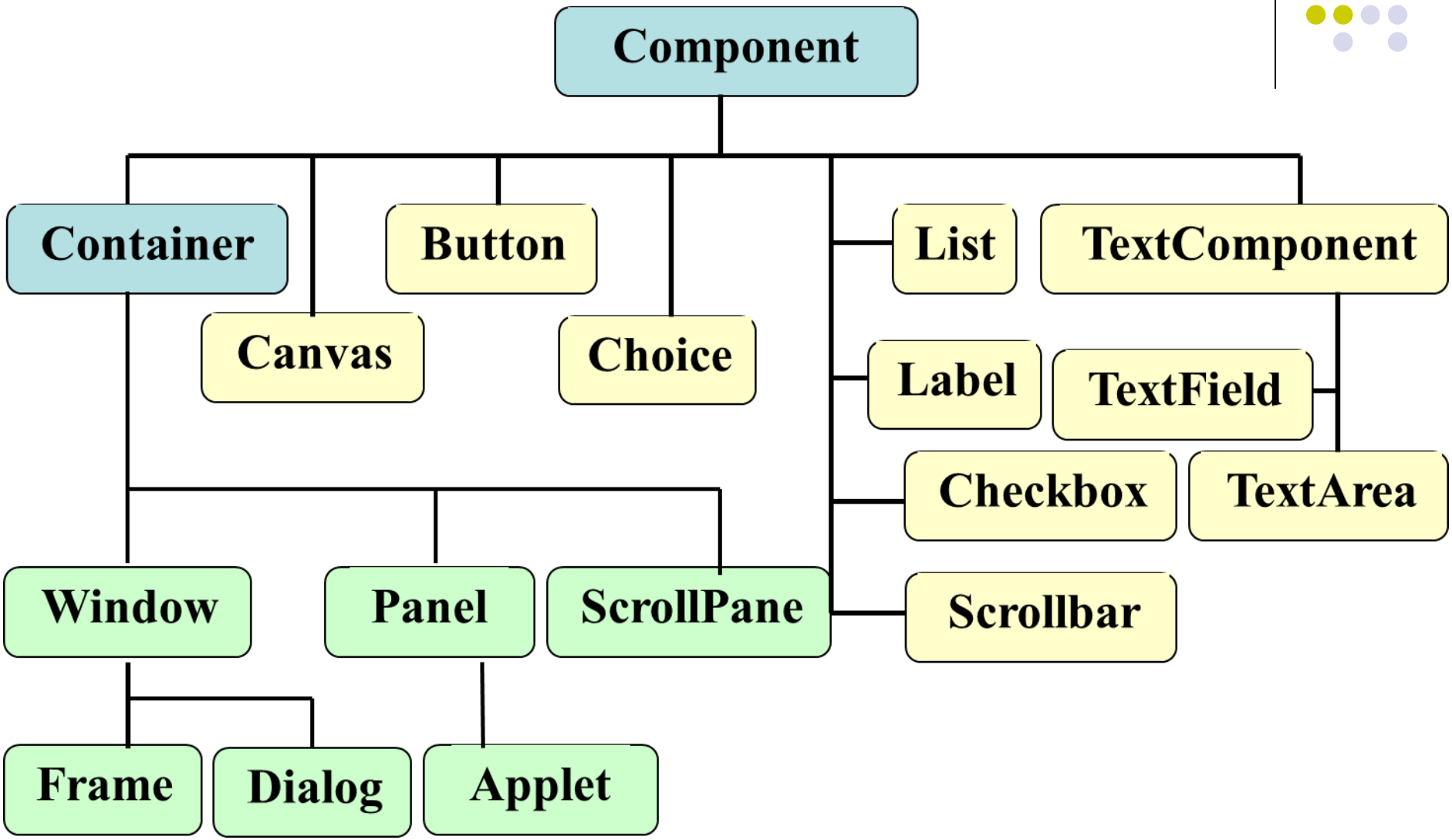


- **AWT包含四个主要的类：组件类(Component)、容器类(Container)、图形类(Graphics)和布局管理器类(LayoutManager和LayoutManager2)。**
  - **Component(组件)类——菜单、按键、列表等组件的抽象基本类。**
  - **Container(容器)类——扩展Component的抽象基本类。由Container派生的类有Panel、Applet、Window、Dialog和Frame类等。在容器中，可以包含多个组件。**
  - **Graphics(图形类)类——定义组件内图形操作的基本类。每个组件都有一个相关的图形对象。**
  - **LayoutManager(布局管理器)类——定义容器中组件的位置和尺寸的接口。Java中定义了几种默认的布局管理器。**

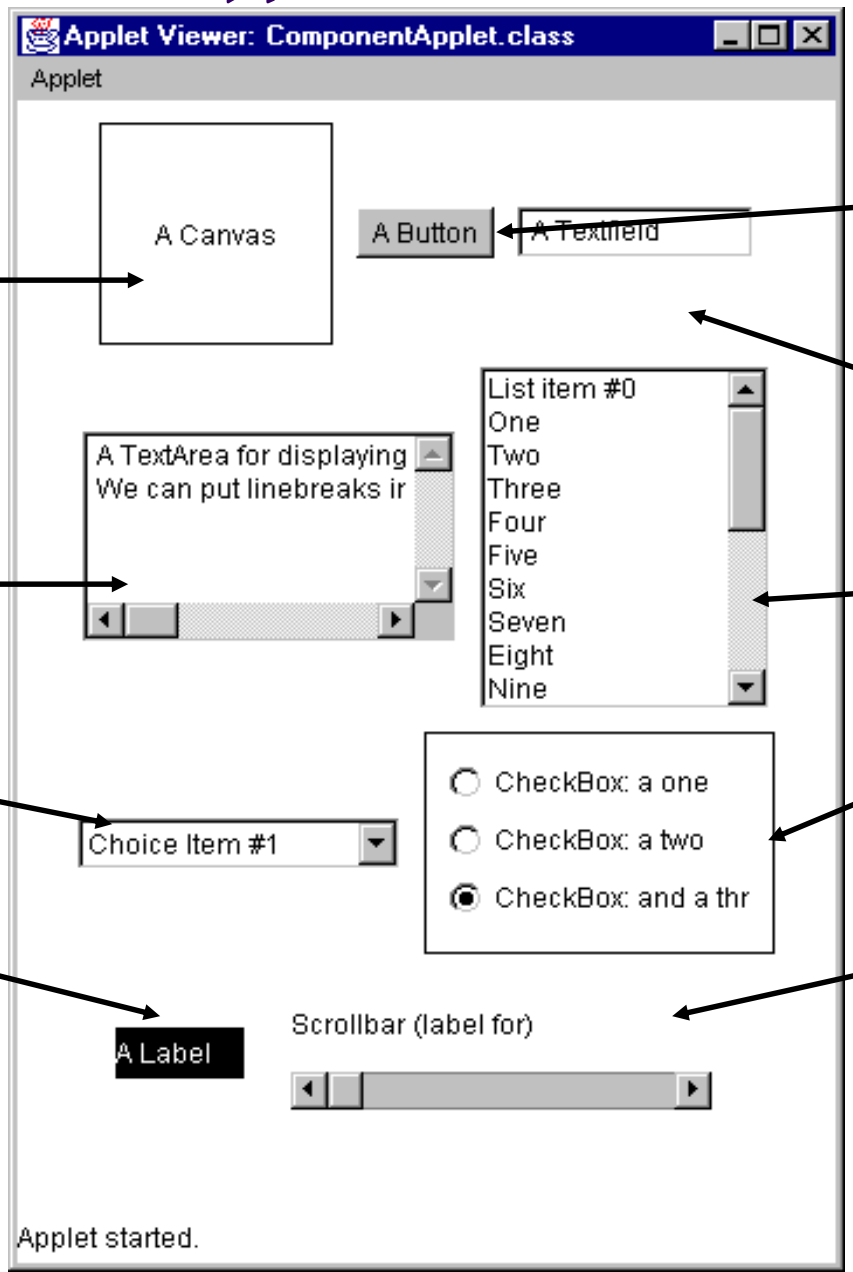
AWT包主要类的层次关系



AWT包主要类的层次关系



8.2.2 AWT组件



Canvas

Button

TextField

TextArea

List

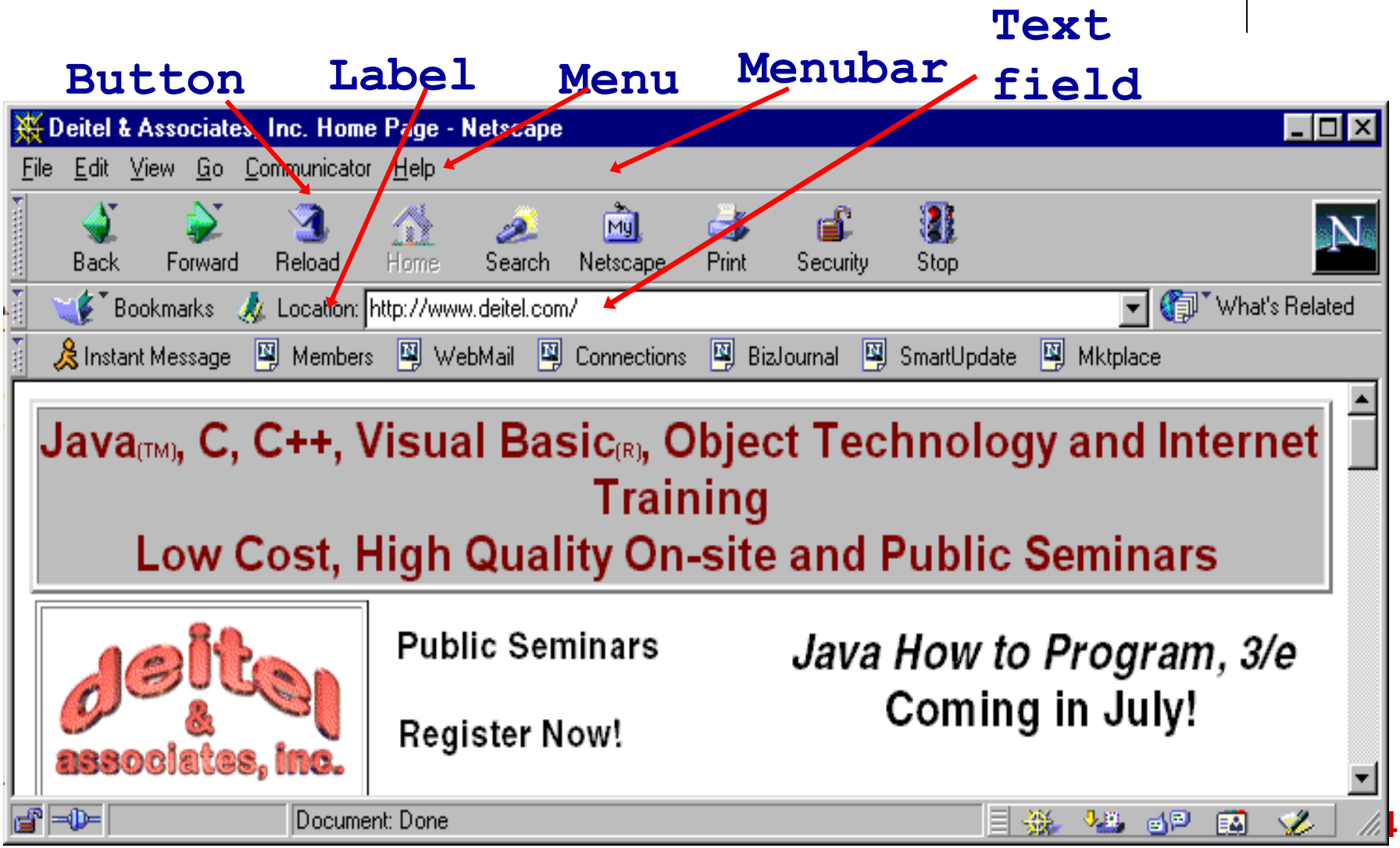
Choice

CheckBox

Label

ScrollBar

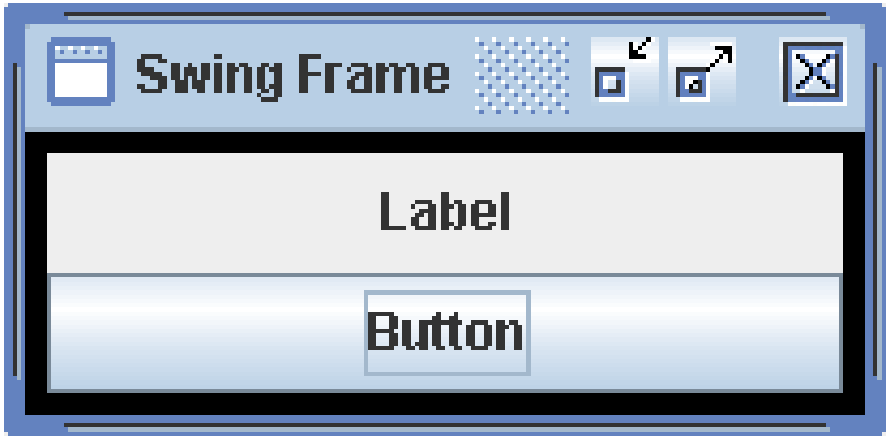
8.2.2 AWT组件


Button
Label
Menu
Menubar
Text field
Deitel & Associates, Inc. Home Page - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: http://www.deitel.com/ What's Related
Instant Message Members WebMail Connections BizJournal SmartUpdate Mktplace
Java(TM), C, C++, Visual Basic(R), Object Technology and Internet Training
Low Cost, High Quality On-site and Public Seminars
deitel & associates, inc.
Public Seminars
Register Now!
Java How to Program, 3/e
Coming in July!
Document: Done

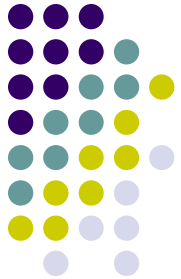
8.2.3 AWT容器

- 容器:是一种可以含有其他组件的组件，AWT中的容器都是从Container抽象类派生而来的.
- 类型：Window,panel,ScrollPane
- 常用：Panel,Frame,Applet
- Panel和Applet：Applet从Panel类继承而来，而 Panel 从Container类继承而来，它不创建自己的窗口，因为它常用于将组件编组放入其它容器(Frame,Applet)中,它缺省的布局管理器为FlowLayout。
- Frame：是一个功能齐全的、顶层的、可重定义尺寸的、带有菜单条的窗口。可以指定标题、图标和光标。它缺省的布局管理器为BorderLayout，且生成与窗口一样的事件：WindowOpened,WindowClosing,WindowClosed,…

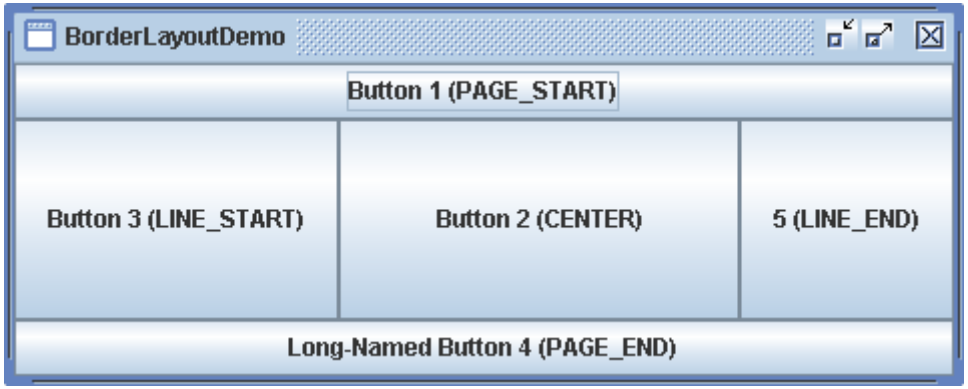
8.2.3 AWT容器



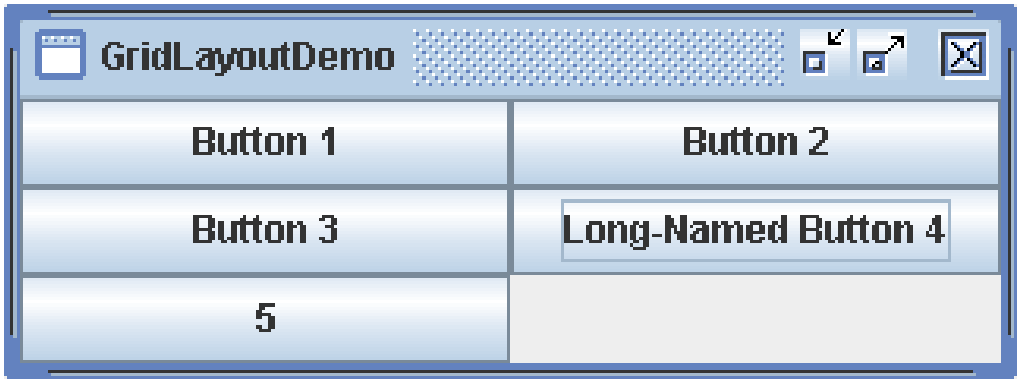
8.2.4 AWT布局管理

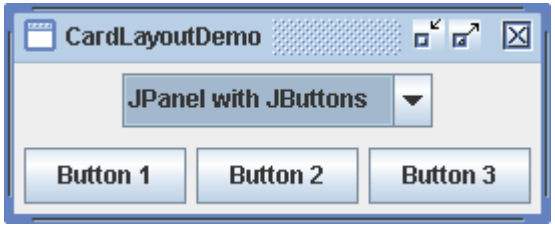


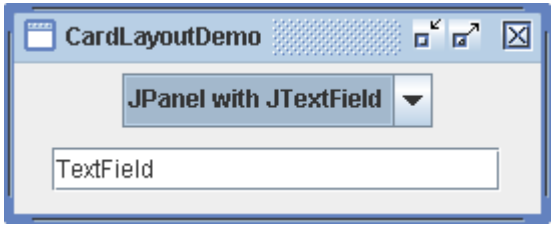
- 常用布局管理器
  - **FlowLayout：组件从左到右、从上到下，一个挨一个地放在容器中。Panel和 Applet的默认容器。**
  - **GridLayout：网格布局管理器。每个网格单元放置一个组件或容器。**
  - **BorderLayout：按照东、西、南、北、中安排组件。是Window、Frame、Dialog的默认容器。**
  - **CardLayout：卡式布局管理器。**
  - **GridBagLayout：复杂的网格布局管理器。**


BorderLayoutDemo
Button 1 (PAGE_START)
Button 3 (LINE_START)
Button 2 (CENTER)
5 (LINE_END)
Long-Named Button 4 (PAGE_END)


FlowLayoutDemo
Button 1
Button 2
Button 3
Long-Named Button 4
5


GridLayoutDemo
Button 1
Button 2
Button 3
Long-Named Button 4
5


CardLayoutDemo
JPanel with JButtons
Button 1
Button 2
Button 3


CardLayoutDemo
JPanel with JTextField
TextField

8.2.5 AWT事件处理

- GUI是由事件驱动的，一些常见的事件包括：
  - 移动鼠标
  - 单双击鼠标各个按钮
  - 单击按钮
  - 在文本字段输入
  - 在菜单中选择菜单项
  - 在组合框中选择、单选和多选
  - 拖动滚动条
  - 关闭窗口
  - ……
- AWT通过事件对象来包装事件，程序可以通过事件对象获得事件的有关信息
- 事件是通过事件监听器(event listeners)来管理的.
- 源码顶部需加入：**import java.awt.event.***