第12章
Introduction to Java Programming, 8E

图形用户界面基础

学习目标

- 区分Swing和AWT的不同（12.2节）。
- 描述Java GUI API的层次体系结构（12.3节）。
- 使用框架、面板和简单GUI组件创建用户界面（12.4节）。
- 理解布局管理器的作用（12.5节）。
- 使用`FlowLayout`、`GridLayout`和`BorderLayout`管理器在一个容器中布局组件（12.5节）。
- 使用JPanel类将面板作为一个子容器（12.6节）。
- 使用`Color`类和`Font`类指定颜色和字体（12.7～12.8节）。
- 将边界、工具提示、字体和颜色等常用特性应用在Swing组件上（12.9节）。
- 使用边界可视化地将用户界面组件分组（12.9节）。
- 使用`ImageIcon`类创建图形图标（12.10节）。

12.1 引言

为Java GUI程序设计而设计的API是应用面向对象原理的绝佳范例。本章有两个目的。第一，它介绍了Java GUI程序设计的基础知识。第二，它使用GUI演示面向对象程序设计。尤其是本章还将介绍Java GUI API的框架结构，还要讨论GUI组件以及组件之间的相互关系、容器和布局管理器、颜色、字体、边界、图像图标以及工具提示。

12.2 Swing和AWT

在8.6.3节中使用了简单的GUI例子演示OOP。我们已经使用过像`JButton`、`JLabel`、`JTextField`、`JRadioButton`和`JComboBox`这样的GUI组件。为什么GUI组件的类名都有前缀*J*呢？为什么不是简单地将它命名为`Button`，而是使用`JButton`来命名呢？事实上，在包`java.awt`中已经有一个名为`Button`的类。

介绍Java的时候，将图形用户界面相关的类捆绑在一起，放在一个称为抽象窗口工具箱（Abstract Window Toolkit，AWT）的库中。AWT适合开发简单的图形用户界面，但并不适合开发复杂的GUI项目。除此之外，AWT更容易发生与特定平台相关的故障。AWT的用户界面组件就被一种更稳定、更通用和更灵活的库取代，这种库称为Swing组件（Swing component）库。大多数Swing组件都是直接用Java代码在画布上绘图的，而`java.awt.Window`或`java.awt.Panel`的子类的组件例外，它们必须使用特定平台上自己的GUI来绘图。Swing组件更少地依赖于目标平台并且更少地使用自己的GUI资源。因此，不依赖于自己GUI的Swing组件称为轻量级组件（lightweight component），而AWT组件称为重量级组件（heavyweight component）。

为了区别新的Swing组件类和与它对应的AWT组件类，Swing GUI组件类都以字母*J*为前缀来命名。尽管在Java中仍然支持AWT组件，但是最好学习如何使用Swing组件编程，因为AWT用户界面组件终究是要退出历史舞台的。本书只使用Swing GUI组件。

12.3 Java GUI API

GUI API包含的类可以分成三个组：组件类（component class）、容器类（container class）和辅助类（helper class）。它们的层次体系结构关系如图12-1所示。

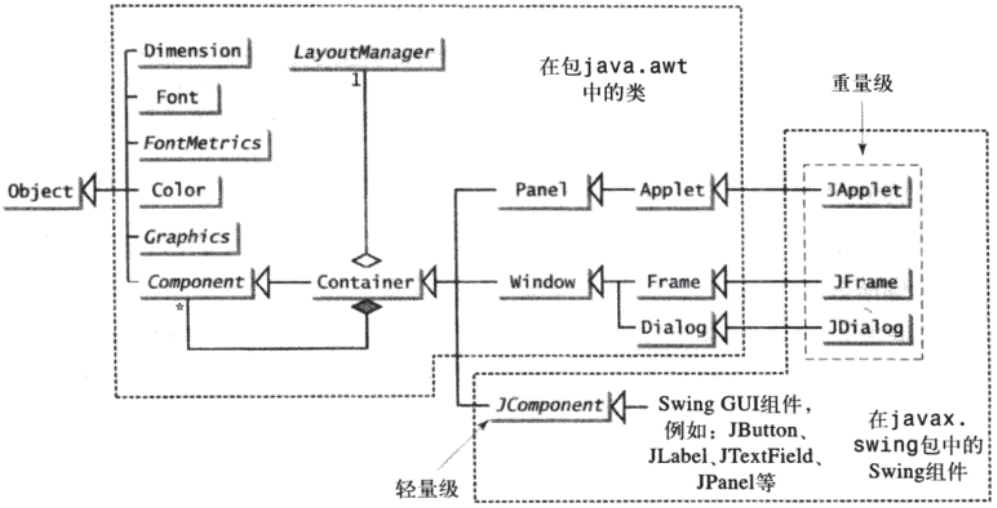


图12-1 Java GUI程序设计使用的是这个体系结构图中所展示的类

组件类是用来创建用户界面的，例如，JButton、JLabel和JTextField。容器类是用来包含其他组件的，例如，JFrame、JPanel和JApplet。辅助类是用来支持GUI组件的，例如，Graphics、Color、Font、FontMetrics和Dimension。

注意 JFrame、JApplet、JDialog和JComponent类及其子类一起放置在javax.swing包中。图12-1中的其他类均一起放置在java.awt包中。

12.3.1 组件类

Component类的实例可以显示在屏幕上。Component类是包括容器类的所有用户界面类的根类，而JComponent类是所有轻量级Swing组件类的根类。Component和JComponent都是抽象类。抽象类将在第14章中介绍。当目前为止，只需要知道抽象类除了不能使用new操作符创建它的实例之外，其他地方和其他类是一样的。例如，不能使用new JComponent()创建一个JComponent的实例。但是，可以使用JComponent的具体子类的构造方法来创建JComponent的实例。熟悉这些类的继承层次结构是非常必要的。例如，下面语句的结果都显示true：

```
JButton jbtOK = new JButton("OK");
System.out.println(jbtOK instanceof JButton);
System.out.println(jbtOK instanceof JComponent);
System.out.println(jbtOK instanceof Container);
System.out.println(jbtOK instanceof Component);
System.out.println(jbtOK instanceof Object);
```

12.3.2 容器类

一个Container的实例可以包含Component实例。容器类是用于盛装其他GUI组件的GUI组件。Window、Panel、Applet、Frame和Dialog都是AWT组件的容器类。要使用Swing组件作容器，可以使用表12-1中所描述的Container、JFrame、JDialog、JApplet和JPanel。

表12-1 GUI容器类

容器类	说 明
java.awt.Container	用于对组件分组。框架Frame、面板Panel和applet都是它的子类
javax.swing.JFrame	一个不能包含在另一个窗口中的窗口。在Java GUI应用程序中，它用于存放其他Swing用户界面组件
java.swing.JPanel	一个存放用户界面组件的不可见的容器。面板可以嵌套。可以将面板放在包含面板的容器中。JPanel也可用作画图的画布
java.swing.JApplet	Applet的一个子类。必须扩展JApplet才能创建基于Swing的Java applet
java.swing.JDialog	一个弹出式窗口或消息框，一般用作接收来自用户的附加信息或通知事件发生的临时窗口

12.3.3 GUI辅助类

辅助类都不是Component的子类，例如，Graphics、Color、Font、FontMetrics、Dimension和LayoutManager等。它们用来描述GUI组件的属性，例如，图形的内容、颜色、字体以及大小尺寸等，如表12-2所示。

表12-2 GUI辅助类

辅助类	说 明
java.awt.Graphics	一个抽象类，提供绘制字符串、线和简单几何图形的方法
java.awt.Color	处理GUI组件的颜色。例如，可以在像JFrame和JPanel这样的组件中指定背景色或前景色，或者指定绘制的线条、几何图形和字符串的颜色
java.awt.Font	指定GUI组件上文本和图形的字体。例如，可以指定按钮上文本的字型（例如，SansSerif）、风格（例如，粗体）以及大小（例如，24号）
java.awt.FontMetrics	一个获取字体属性的抽象类
java.awt.Dimension	将组件的宽度和高度（以整数为精度）封装在单个对象中
java.awt.LayoutManager	指定组件在容器中如何放置

注意 辅助类是在包java.awt中的。Swing组件不能取代AWT中的全部类，只能替代AWT GUI的组件类（例如，Button、TextField、TextArea）。AWT辅助类在GUI程序设计中仍然很有用。

12.4 框架

创建一个用户界面需要创建一个框架或一个applet来存放用户界面组件。在第18章中将介绍Java applet的创建。本节先介绍框架。

12.4.1 创建一个框架

使用JFrame类创建一个框架，如图12-2所示。

程序清单12-1创建了一个框架。

程序清单12-1 MyFrame.java

```
1 import javax.swing.JFrame;
2
3 public class MyFrame {
4   public static void main(String[] args) {
```

```
5     JFrame frame = new JFrame("MyFrame"); // Create a frame
6     frame.setSize(400, 300); // Set the frame size
7     frame.setLocationRelativeTo(null); // Center a frame
8     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
9     frame.setVisible(true); // Display the frame
10  }
11 }
```

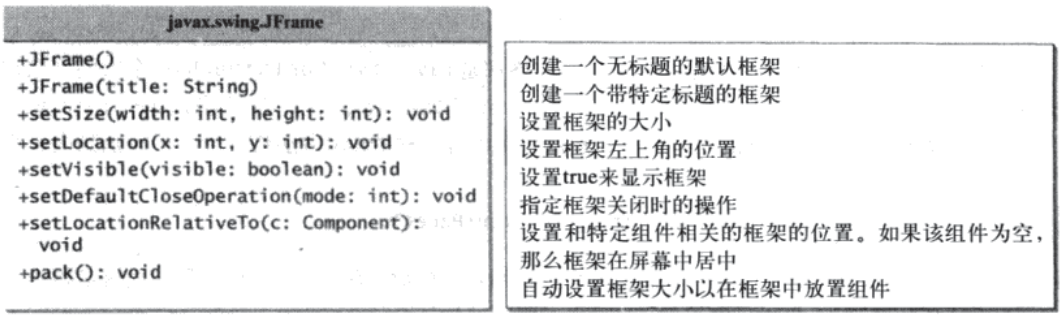


图12-2 JFrame是存放GUI组件的最高层的容器

直到调用frame.setVisible(true)方法之后才会显示框架。frame.setSize(400,300)指定框架的宽度为400像素，高度为300像素。如果没有使用setSize方法，那么框架的大小就只够显示标题栏。由于setSize和setVisible方法都被定义在Component类中，所以，它们都可以被JFrame类所继承。随后还将看到，这些方法在Component的很多其他子类中也是非常有用的。

运行程序MyFrame时，屏幕上就会显示一个窗口（参见图12-3a）。

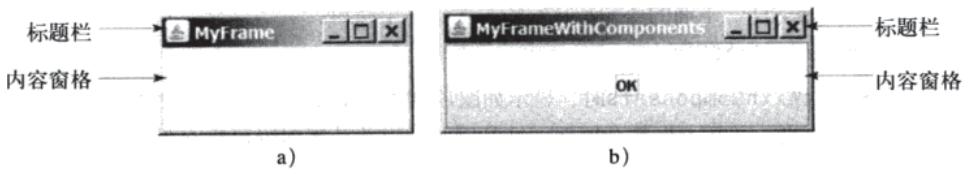


图12-3 a）程序创建并显示一个标题为MyFrame的框架；b）给这个框架添加一个OK按钮

调用setLocationRelativeTo(null)（第7行）方法可以在屏幕上居中显示框架。调用setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE)（第8行）方法来告诉程序，当框架关闭时结束程序。如果不使用这条语句，就会造成关闭框架后程序并不会结束。这种情况下，就必须在Windows系统中的DOS提示符窗口中按下Ctrl+C组合键或者在UNIX系统中使用kill命令来结束这个进程。

注意 回顾一下，像素是在屏幕上绘图的最小空间单位。可以将像素想象成一个小长方形，将屏幕认为是由像素铺砌而成的。分辨率表示每平方英寸的像素数。屏幕像素越多，屏幕的分辨率就越高。分辨率越高，可以看到的细节越多。

注意 应该在调用setLocationRelativeTo(null)之前调用setSize(w,h)将框架居中。

12.4.2 向框架中添加组件

图12-3a显示的框架是空的。可以使用add方法在框架中添加组件，如程序清单12-2所示。

程序清单12-2 MyFrameWithComponents.java

```
1 import javax.swing.*;
2
3 public class MyFrameWithComponents {
4   public static void main(String[] args) {
5     JFrame frame = new JFrame("MyFrameWithComponents");
6
```

```
 7     // Add a button into the frame
 8     JButton jbtOK = new JButton("OK");
 9     frame.add(jbtOK);
10
11     frame.setSize(400, 300);
12     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
13     frame.setLocationRelativeTo(null); // Center the frame
14     frame.setVisible(true);
15   }
16 }
```

每个JFrame都包含一个内容窗格。每个内容窗格都是java.awt.Container的一个实例。像按钮之类的GUI组件放置在框架的内容窗格中。在Java早期的版本中，必须使用JFrame类中的getContentPane方法返回框架的内容窗格，然后调用内容窗格的add方法将一个组件添加到内容窗格中，如下所示：

```
java.awt.Container container = frame.getContentPane();
container.add(jbtOK);
```

这是很麻烦的。所以，Java 5之后的Java新版本允许调用框架的add方法，将组件放置在内容窗格中，如下所示：

```
frame.add(jbtOK);
```

这种新特性称为内容窗格授权（content-pane delegation）。严格地讲，就是将一个组件添加到框架的内容窗格中。为简便起见，我们就说将组件添加到框架中。

使用new JButton("OK")创建一个JButton对象，并把该对象添加到框架的内容窗格中（第9行）。

定义在Container类中的add(Component comp)方法给容器中添加一个Component实例。由于JButton是Component的一个子类，所以，JButton的实例也是Component的实例。为了从容器中删除组件，可以使用remove方法。下面的语句是从容器中删除一个按钮：

```
container.remove(jbtOK);
```

运行程序MyFrameWithComponents时，显示如图12-3b所示的窗口。不管如何调整窗口的大小，按钮都会显示在框架的中央，并且占据整个框架。这是因为，组件是被内容窗格的布局管理器放置在框架上的，而内容窗格的默认布局管理器就是将按钮放到中央。在下一节中，将会使用几种不同的布局管理器将组件放置在需要的位置上。

12.5 布局管理器

在许多其他窗口系统中，用户界面组件是通过使用硬编码（hard-code）的像素度量管理的。例如，将一个按钮放在窗口的（10，10）位置处。使用硬编码的像素度量，这个用户界面可能在一个系统中看上去很好，但在另一个系统上就不正常。Java的布局管理器提供了一种层面的抽象，自动将用户界面映射到所有的窗口系统。

Java的GUI组件都放置在容器中，它们的位置是由容器的布局管理器来管理的。在前面的程序中，并没有指定将OK按钮放置在框架中的什么位置，但是，Java知道应该把它放在哪里，因为在后台工作的布局管理器能够将组件放到正确的位置。布局管理器是使用布局管理器类创建的。

使用setLayout(aLayoutManager)方法在容器中设置布局管理器。例如，可以使用下面的语句创建一个XLayout的实例，并将它置于一个容器内：

```
LayoutManager layoutManager = new XLayout();
container.setLayout(layoutManager);
```

本节介绍三种基本的布局管理器：FlowLayout、GridLayout和BorderLayout。

12.5.1 FlowLayout

FlowLayout是最简单的布局管理器。按照组件添加的顺序，从左到右地将组件排列在容器中。当放满一行时，就开始新的一行。可以使用三个常量FlowLayout.RIGHT、FlowLayout.CENTER和FlowLayout.LEFT之一来指定组件的对齐方式。还可以指定组件之间以像素为单位的间隔。FlowLayout的类图如图12-4所示。

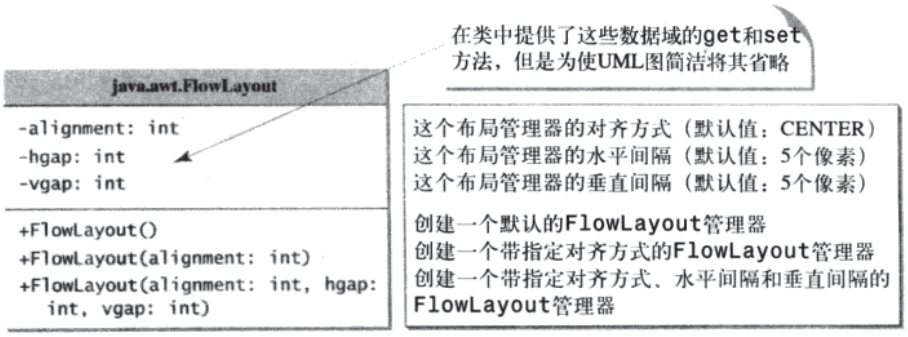


图12-4 FlowLayout逐行放置组件

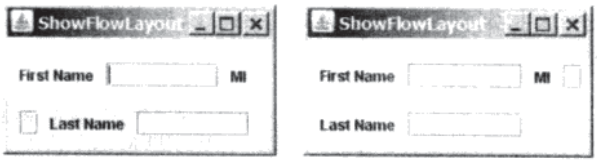
程序清单12-3给出一个演示流布局的程序。这个程序使用FlowLayout管理器向这个框架添加三个标签和文本域，如图12-5所示。

程序清单12-3 ShowFlowLayout.java

```
 1 import javax.swing.JLabel;
 2 import javax.swing.JTextField;
 3 import javax.swing.JFrame;
 4 import java.awt.FlowLayout;
 5
 6 public class ShowFlowLayout extends JFrame {
 7   public ShowFlowLayout() {
 8     // Set FlowLayout, aligned left with horizontal gap 10
 9     // and vertical gap 20 between components
10     setLayout(new FlowLayout(FlowLayout.LEFT, 10, 20));
11
12     // Add labels and text fields to the frame
13     add(new JLabel("First Name"));
14     add(new JTextField(8));
15     add(new JLabel("MI"));
16     add(new JTextField(1));
17     add(new JLabel("Last Name"));
18     add(new JTextField(8));
19   }
20
21   /** Main method */
22   public static void main(String[] args) {
23     ShowFlowLayout frame = new ShowFlowLayout();
24     frame.setTitle("ShowFlowLayout");
25     frame.setSize(200, 200);
26     frame.setLocationRelativeTo(null); // Center the frame
27     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
28     frame.setVisible(true);
29   }
30 }
```

这个例子使用和12.4节程序不同的风格创建一个程序，12.4节程序的框架是用JFrame类创建的。这个例子扩展了JFrame类，创建一个名为ShowFlowLayout的类（第6行）。这个程序中的main方法创建了一个ShowFlowLayout的实例（第23行）。ShowFlowLayout的构造方法在框架中创建并放置组件。这是创建GUI应用程序时推崇的风格，原因有以下三点：

- 创建一个GUI应用程序意味着创建一个框架，所以，会很自然地扩展JFrame类来定义一个框架。
- 这个框架可能会进一步扩展以添加新的组件或者功能。
- 这个类可以很容易地重用。例如，可以通过创建该类的多个实例来创建多个框架。



a) b)

图12-5 组件被FlowLayout管理器逐个地添加到容器的每一行

持续使用一种风格可以使得程序易于阅读。从现在开始，大多数GUI主类都将扩展JFrame类。这个主类的构造方法创建用户界面。main方法创建这个主类的实例，然后显示这个框架。

在这个例子中，使用FlowLayout管理器在框架中放置组件。如果改变框架的大小，组件会自动地重新排列以适合框架。在图12-5a中，第一行有三个组件，但是，在图12-5b中，第一行有四个组件，这是因为宽度增加了。

如果使用setLayout(new FlowLayout(FlowLayout.RIGHT,0,0))代替setLayout语句（第10行），那么所有的按钮行都将右对齐且没有间隔。

在下面的语句中创建了一个匿名的FlowLayout对象（第10行）：

```
setLayout(new FlowLayout(FlowLayout.LEFT, 10, 20));
```

它等价于语句：

```
FlowLayout layout = new FlowLayout(FlowLayout.LEFT, 10, 20);
setLayout(layout);
```

这个代码创建了一个对FlowLayout类的对象layout的显式引用。这个显式引用是没有必要的，因为该对象在ShowFlowLayout类中并不是直接引用的。

假设将同一个按钮在框架中添加10次，那么框架中会出现10个按钮吗？答案是不会，像按钮这样的GUI组件只可以添加到一个容器中，且只能在一个容器中出现一次。将一个按钮向容器添加多次和添加一次是一样的。

警告 在设置布局风格时，不要忘记在布局管理器类之前使用new操作符。例如，setLayout(new FlowLayout())。

注意 构造方法ShowFlowLayout()没有显式地调用构造方法JFrame()，但是构造方法JFrame()被隐式调用。参见11.3.2节。

12.5.2 GridLayout

GridLayout管理器以网格（矩阵）的形式管理组件。组件按照它们添加的顺序从左到右排列，先是第一行，接着是第二行，依此类推。布局管理器GridLayout的类图如图12-6所示。

可以指定网格中的行数和列数。基本规则如下：

1）行数或列数可以为零，但不能两者都为零。如果一个为零另一个不为零，那么不为零的行或列的大小已经固定，而为零的行或列的大小由布局管理器动态地决定。例如，如果指定一个网格有0行3列10个组件，GridLayout就会创建3个固定的列和4个行，最后1行只包含1个组件。如果指定一个网格有3行0列10个组件，GridLayout就会创建3个固定的行和4个列，最后1行包含2个组件。

2）如果行数和列数都不为零，那么行数就是主导参数；也就是说，行数是固定的，布局管理器会动态地计算列数。例如，如果指定一个网格有3行3列10个组件，GridLayout就会创建3个固定的行和4个列，最后1行包含2个组件。

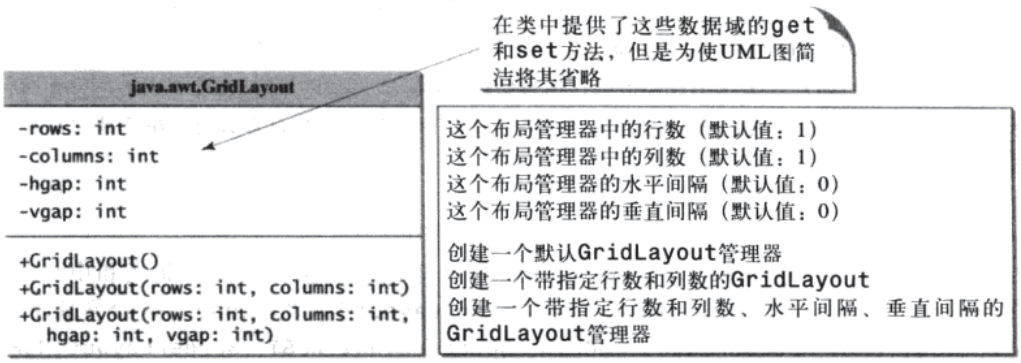


图12-6 GridLayout将组件放在网格上大小相同的单元中

程序清单12-4给出一个演示网格布局的程序。该程序类似于程序清单12-3中的程序。它添加三个标签和三个文本域到GridLayout的框架而不是FlowLayout的框架，如图12-7所示。

程序清单12-4 ShowGridLayout.java

```
 1 import javax.swing.JLabel;
 2 import javax.swing.JTextField;
 3 import javax.swing.JFrame;
 4 import java.awt.GridLayout;
 5
 6 public class ShowGridLayout extends JFrame {
 7   public ShowGridLayout() {
 8     // Set GridLayout, 3 rows, 2 columns, and gaps 5 between
 9     // components horizontally and vertically
10     setLayout(new GridLayout(3, 2, 5, 5));
11
12     // Add labels and text fields to the frame
13     add(new JLabel("First Name"));
14     add(new JTextField(8));
15     add(new JLabel("MI"));
16     add(new JTextField(1));
17     add(new JLabel("Last Name"));
18     add(new JTextField(8));
19   }
20
21   /** Main method */
22   public static void main(String[] args) {
23     ShowGridLayout frame = new ShowGridLayout();
24     frame.setTitle("ShowGridLayout");
25     frame.setSize(200, 125);
26     frame.setLocationRelativeTo(null); // Center the frame
27     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
28     frame.setVisible(true);
29   }
30 }
```

如果改变这个框架的大小，那么按钮的布局保持不变（也就是说，行数和列数不变，间隔也不变）。在GridLayout的容器中，所有组件的大小都被认为是一样的。

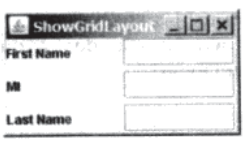


图12-7 GridLayout管理器将容器分为网格，然后添加组件逐行地填充每个格子

使用setLayout(new GridLayout(3,10))代替setLayout语句（第10行）还是会得到3行2列。

因为行的参数非零，所以列的参数被忽略。列的实际数量是由布局管理器计算出来的。

如果用setLayout(new GridLayout(4,2))或者用setLayout(new GridLayout(2,2))代替setLayout语句（第10行），会发生什么情况？请自己试一试。

注意 在FlowLayout和GridLayout两个布局管理器中，组件添加到容器的顺序是很重要的。它决定了组件在容器中的位置。

12.5.3 BorderLayout

BorderLayout管理器将容器分成五个区域：东区、南区、西区、北区和中央。使用add(Component,index)方法可以将组件添加到BorderLayout中，其中index是一个常量，取值为BorderLayout.EAST、BorderLayout.SOUTH、BorderLayout.WEST、BorderLayout.NORTH或BorderLayout.CENTER。BorderLayout的类图如图12-8所示。

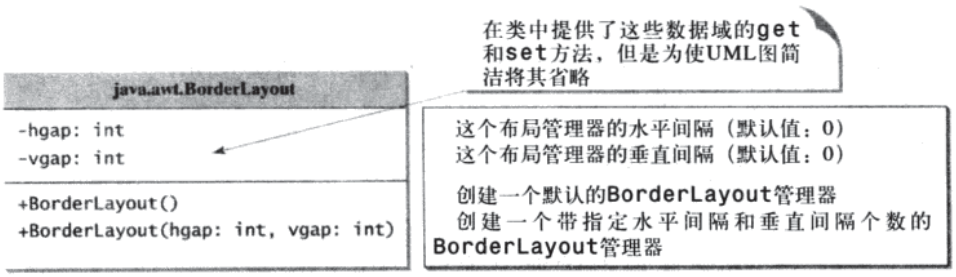


图12-8 BorderLayout将组件放置在五个区域中

组件根据它们最合适的尺寸和它们在容器中的位置来放置。南、北组件可以水平拉伸；东、西组件可以垂直拉伸；中央组件既可以水平拉伸也可以竖直拉伸以填充空白空间。

程序清单12-5给出一个演示边界布局的程序。程序将5个标有East、South、West、North和Center的按钮添加到一个BorderLayout管理器的框架中，如图12-9所示。

程序清单12-5 ShowBorderLayout.java

```
 1 import javax.swing.JButton;
 2 import javax.swing.JFrame;
 3 import java.awt.BorderLayout;
 4
 5 public class ShowBorderLayout extends JFrame {
 6   public ShowBorderLayout() {
 7     // Set BorderLayout with horizontal gap 5 and vertical gap 10
 8     setLayout(new BorderLayout(5, 10));
 9
10     // Add buttons to the frame
11     add(new JButton("East"), BorderLayout.EAST);
12     add(new JButton("South"), BorderLayout.SOUTH);
13     add(new JButton("West"), BorderLayout.WEST);
14     add(new JButton("North"), BorderLayout.NORTH);
15     add(new JButton("Center"), BorderLayout.CENTER);
16   }
17
18   /** Main method */
19   public static void main(String[] args) {
20     ShowBorderLayout frame = new ShowBorderLayout();
21     frame.setTitle("ShowBorderLayout");
22     frame.setSize(300, 200);
23     frame.setLocationRelativeTo(null); // Center the frame
24     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
25     frame.setVisible(true);
26   }
27 }
```

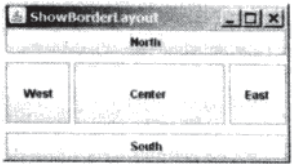


图12-9 BorderLayout将容器分为五个区域，每个区域都可以包含一个组件

将按钮添加到框架中（第11～15行）。注意，BorderLayout的add方法与FlowLayout和GridLayout的add方法不同，使用BorderLayout管理器指定组件放置的位置。

没有必要将组件放得占满整个区域。如果在程序中删掉东区按钮并重新运行它，将会发现中央按钮向右延伸占据东区。

注意 BorderLayout将省略的下标说明解释为BorderLayout.CENTER。例如，add(component)和add(Component,BorderLayout.CENTER)是一样的。如果要在BorderLayout的容器中添加两个组件，如下所示：

```
container.add(component1);
container.add(component2);
```

只会显示最后一个组件。

12.5.4 布局管理器的属性

可以动态地改变布局管理器的属性。FlowLayout具有属性alignment、hgap和vgap。可以使用setAlignment、setHgap和setVgap方法来表明对齐方式、水平间隔和垂直间隔。GridLayout具有属性rows、columns、hgap和vgap。可以使用setRows、setColumns、setHgap和setVgap方法来指定行数、列数以及水平间隔和垂直间隔。BorderLayout具有属性hgap和vgap。可以使用setHgap和setVgap方法来指定水平间隔和垂直间隔。

在前几节中，因为一旦创建了布局管理器，它的属性就不能改变，所以使用的都是匿名布局管理器。如果需要动态地改变布局管理器的属性，布局管理器必须用一个变量显式地引用。然后，可以通过这个变量来改变布局管理器的属性。例如，下面的代码创建一个布局管理器并且设置它的属性：

```
// Create a layout manager
FlowLayout flowLayout = new FlowLayout();

// Set layout properties
flowLayout.setAlignment(FlowLayout.RIGHT);
flowLayout.setHgap(10);
flowLayout.setVgap(20);
```

12.6 使用面板作为子容器

假设要在框架中放置十个按钮和一个文本域。按钮以网格形式放置，文本域单独占一行。如果将所有这些组件放在一个单独的容器中，是很难达到要求的视觉效果的。使用Java图形用户界面进行程序设计，可以将一个窗口分成几个面板。面板的作用就是分组放置用户界面组件的子容器。可以将这些按钮添加到一个面板中，然后再将这个面板添加到框架中。

面板的Swing版本是JPanel。可以使用new JPanel()创建一个带默认FlowLayout管理器的面板，也可以使用new JPanel(LayoutManager)创建一个带特定布局管理器的面板。使用add(Component)方法可以向面板添加一个组件。例如，下面的代码创建了一个面板并且给它添加一个按钮：

```
JPanel p = new JPanel();
p.add(new JButton("OK"));
```

面板可以放到一个框架中或者放在另一个面板中。下面的语句将面板p放到框架f中：

```
f.add(p);
```

程序清单12-6是一个演示使用面板作为子容器的例子。程序创建了一个微波炉的用户界面，如图12-10所示。

程序清单12-6 TestPanels.java

```
1 import java.awt.*;
2 import javax.swing.*;
3
4 public class TestPanels extends JFrame {
5   public TestPanels() {
6     // Create panel p1 for the buttons and set GridLayout
7     JPanel p1 = new JPanel();
8     p1.setLayout(new GridLayout(4, 3));
9
10    // Add buttons to the panel
11    for (int i = 1; i <= 9; i++) {
12      p1.add(new JButton("" + i));
13    }
14
15    p1.add(new JButton("" + 0));
16    p1.add(new JButton("Start"));
17    p1.add(new JButton("Stop"));
18
19    // Create panel p2 to hold a text field and p1
20    JPanel p2 = new JPanel(new BorderLayout());
21    p2.add(new JTextField("Time to be displayed here"),
22      BorderLayout.NORTH);
23    p2.add(p1, BorderLayout.CENTER);
24
25    // add contents into the frame
26    add(p2, BorderLayout.EAST);
27    add(new JButton("Food to be placed here"),
28      BorderLayout.CENTER);
29  }
30
31  /** Main method */
32  public static void main(String[] args) {
33    TestPanels frame = new TestPanels();
34    frame.setTitle("The Front View of a Microwave Oven");
35    frame.setSize(400, 250);
36    frame.setLocationRelativeTo(null); // Center the frame
37    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
38    frame.setVisible(true);
39  }
40 }
```

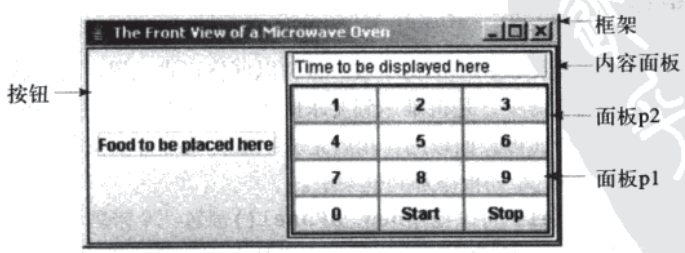


图12-10 程序使用面板组织组件

setLayout方法是在java.awt.Container中定义的。由于JPanel是Container的子类，所以，可以使用setLayout在面板中设置一个新的布局管理器（第8行）。第7～8行可以用语句JPanel p1 =

new JPanel(new GridLayout(4,3))代替。

为了得到所需的布局，程序使用GridLayout的面板p1将数字按钮、Stop按钮和Start按钮放在一组，使用BorderLayout的面板p2将文本域放在北区，将p1放在中央。表示食物的按钮放在框架的中央，而p2放在框架的东区。

语句（第21～22行）

```
p2.add(new JTextField("Time to be displayed here"),
  BorderLayout.NORTH);
```

创建了一个JTextField的实例，然后将它添加到p2。JTextField是一个GUI组件，可以被用户用作输入和显示值。

注意 Container类是像JButton这样的GUI组件类的超类，这点是很值得注意的。每个GUI组件都是一个容器。理论上讲，可以使用setLayout方法在按钮中设置布局，并且将组件添加到按钮中，因为Container类中所有的公共方法都被继承到JButton，但是，如果不是实际情况不得已，是不应该将按钮作为容器使用的。

12.7 Color类

可以使用java.awt.Color类为GUI组件设置颜色。颜色是由红、绿、蓝这三原色构成的，每种原色都用一个int值表示它的深度，取值范围从0（最暗度）到255（最亮度）。这就是通常所说的RGB模式(RGB model)。

可以使用下面的构造方法创建一个color对象：

```
public Color(int r, int g, int b);
```

其中r、g和b指定某个颜色的红、绿、蓝成分。例如，

```
Color color = new Color(128, 100, 100);
```

注意 参数r、g和b的取值都在0到255之间。如果传递给参数的值超过这个范围，就会导致一个IllegalArgumentException异常。

可以使用定义在java.awt.Component类中的setBackground(Color c)和setForeground (Color c)方法来设置一个组件的背景色和前景色。下面是设置一个按钮背景色和前景色的例子：

```
JButton jbtOK = new JButton("OK");
jbtOK.setBackground(color);
jbtOK.setForeground(new Color(100, 1, 1));
```

还可以选择使用java.awt.Color中定义为常量的13种标准颜色（BLACK黑色、BLUE蓝色、CYAN青色、DARK_GRAY深灰、GRAY灰色、GREEN绿色、LIGHT_GRAY淡灰、MAGENTA洋红、ORANGE橘色、PINK粉红、RED大红、WHITE白色和YELLOW黄色）之一。例如，下面的代码可以将按钮的前景色设置成红色：

```
jbtOK.setForeground(Color.RED);
```

12.8 Font类

可以使用java.awt.Font类创建一种字体，然后使用Component类中的SetFont方法设置组件的字体。

Font的构造方法是：

```
public Font(String name, int style, int size);
```

可以从SansSerif、Serif、Monospaced、Dialog或DialogInput中选择一种字体名，可以从Font.PLAIN(0)、Font.BOLD(1)、Font.ITALIC(2)和Font.BOLD + Font.ITALIC(3)中选择风格，然后指定正整数的字体大小。例如，下面的语句创建两种字体，并且给按钮设置一种字体：

```
Font font1 = new Font("SansSerif", Font.BOLD, 16);
Font font2 = new Font("Serif", Font.BOLD + Font.ITALIC, 12);

JButton jbtOK = new JButton("OK");
jbtOK.setFont(font1);
```

提示 如果系统支持其他字体，例如，“Times New Roman”，那就可以使用它创建一个Font对象。为了找出系统上可用的字体，需要使用java.awt.GraphicsEnvironment类的静态方法getLocalGraphicsEnvironment()创建这个类的一个实例。GraphicsEnvironment是描述特定系统上图形环境的一个抽象类。可以使用它的getAllFonts()方法来获取系统中所有可用的字体，也可以使用它的getAvailableFontFamilyNames()方法来获取所有可用字体的名字。例如，下面的语句打印系统中所有可用字体的名字：

```
GraphicsEnvironment e =
  GraphicsEnvironment.getLocalGraphicsEnvironment();
String[] fontnames = e.getAvailableFontFamilyNames();

for (int i = 0; i < fontnames.length; i++)
  System.out.println(fontnames[i]);
```

12.9 Swing GUI组件的公共特性

在本章中，已经使用了一些GUI组件（例如，JFrame、Container、JPanel、JButton、JLabel和JTextField）。本书还将介绍更多的GUI组件。理解这些Swing GUI组件的一般特性是很重要的。Component类是所有GUI组件和容器的根。所有Swing GUI组件（除了JFrame、JApplet和JDialog）都是JComponent的子类，如图12-1所示。图12-11列出了Component、Container和JComponent中对像字体、颜色、大小、工具提示文本及其边界这样的属性的常用操作方法。

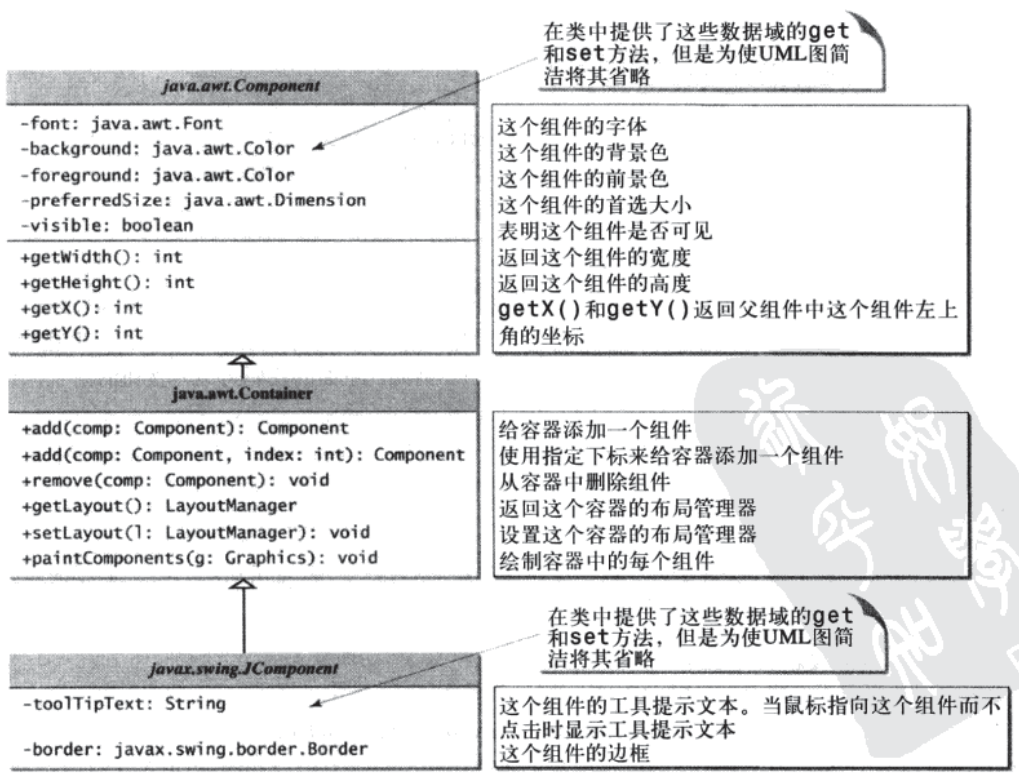


图12-11 所有Swing GUI组件都继承Component、Container和JComponent的公共方法

工具提示（tool tip）是将鼠标移动到组件上时，这个组件上显示的文本。经常用它来描述一个组件的功能。

可以给JComponent类的任何对象设置边界。Swing具有各种类型的边界。为了创建一个带标题的边界，使用new TitleBorder(String title)。为了创建一个线边界，使用new LineBorder (Color color,int width)这里的width表明线的粗细。

程序清单12-7是演示Swing一般特性的例子。该例创建一个面板p1放置三个按钮（第8行），创建p2放置两个标签（第25行），如图12-12所示。按钮jbtLeft的背景色设置为白色（第12行），按钮jbtCenter的前景色设置为绿色（第13行）。按钮jbtRight的工具提示在第14行设置。在面板p1和p2上设置标题边界（第18、36行），而在标签上设置线边界（第32～33行）。

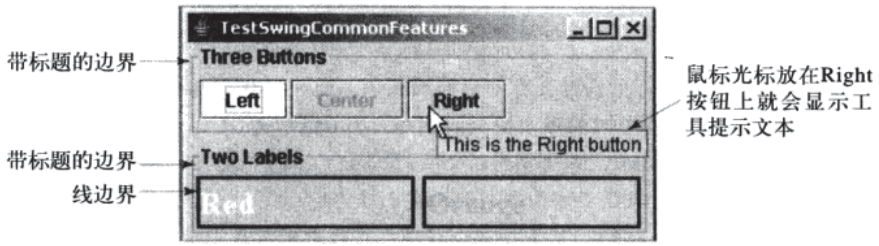


图12-12 在消息面板中设置字体、颜色、边界和工具提示文本

程序清单12-7 TestSwingCommonFeatures.java

```
1  import java.awt.*;
2  import javax.swing.*;
3  import javax.swing.border.*;
4
5  public class TestSwingCommonFeatures extends JFrame {
6    public TestSwingCommonFeatures() {
7      // Create a panel to group three buttons
8      JPanel p1 = new JPanel(new FlowLayout(FlowLayout.LEFT, 2, 2));
9      JButton jbtLeft = new JButton("Left");
10     JButton jbtCenter = new JButton("Center");
11     JButton jbtRight = new JButton("Right");
12     jbtLeft.setBackground(Color.WHITE);
13     jbtCenter.setForeground(Color.GREEN);
14     jbtRight.setToolTipText("This is the Right button");
15     p1.add(jbtLeft);
16     p1.add(jbtCenter);
17     p1.add(jbtRight);
18     p1.setBorder(new TitledBorder("Three Buttons"));
19
20     // Create a font and a line border
21     Font largeFont = new Font("TimesRoman", Font.BOLD, 20);
22     Border lineBorder = new LineBorder(Color.BLACK, 2);
23
24     // Create a panel to group two labels
25     JPanel p2 = new JPanel(new GridLayout(1, 2, 5, 5));
26     JLabel jlblRed = new JLabel("Red");
27     JLabel jlblOrange = new JLabel("Orange");
28     jlblRed.setForeground(Color.RED);
29     jlblOrange.setForeground(Color.ORANGE);
30     jlblRed.setFont(largeFont);
31     jlblOrange.setFont(largeFont);
32     jlblRed.setBorder(lineBorder);
33     jlblOrange.setBorder(lineBorder);
34     p2.add(jlblRed);
35     p2.add(jlblOrange);
36     p2.setBorder(new TitledBorder("Two Labels"));
37
```

```
38      // Add two panels to the frame
39      setLayout(new GridLayout(2, 1, 5, 5));
40      add(p1);
41      add(p2);
42    }
43
44    public static void main(String[] args) {
45      // Create a frame and set its properties
46      JFrame frame = new TestSwingCommonFeatures();
47      frame.setTitle("TestSwingCommonFeatures");
48      frame.setSize(300, 150);
49      frame.setLocationRelativeTo(null); // Center the frame
50      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
51      frame.setVisible(true);
52    }
53 }
```

注意 在不同的组件中同样的属性会有不同的默认值。例如，JFrame中的visible属性默认值为false，但是在JComponent的每个实例（例如，JButton和JLabel）中该属性默认值都为true。为了显示一个JFrame，必须调用setVisible(true)将属性visible设置为true，但是不必为JButton或JLabel设置该属性，因为它已经为true。为使JButton或JLabel不可见，可以调用setVisible(false)。请运行这个程序，然后看看在第37行插入下面两条语句之后的效果：

```
jbtLeft.setVisible(false);
jlblRed.setVisible(false);
```

12.10 图像图标

图标是一个大小固定的图片；通常情况下，它都比较小，用来装饰组件。图像通常存储在图像文件中。Java目前支持三种图像格式：GIF（图像交换格式）、JPEG（联合图像专家组）以及PNG（便携网络图片）。这些类型的图像文件名分别以.gif、.jpg和.png结尾。如果有一个其他格式的位图文件或图像文件，可以使用图像处理工具将它们转为GIF、JPEG或PNG格式，以便于在Java中使用。

为了显示一个图像图标，首先使用new java.swing.ImageIcon(filename)创建一个ImageIcon对象。例如，下面的语句使用当前类路径下的image目录的图像文件us.gif来创建一个图标：

```
ImageIcon icon = new ImageIcon("image/us.gif");
```

“image/us.gif”放置在“c:\book\image\us.gif”下。反斜杠（\）是Windows系统的文件路径符号。在UNIX系统中，应该使用的是斜杠（/）。在Java中，斜杠（/）用来表示在Java的classpath下的相对文件路径（例如，本例中的image/us.gif）。

提示 在Windows系统中，文件名是不区分大小写的，但是在UNIX系统中是区分大小写的。为了使程序可以在所有平台上运行，就将所有的图像文件统一使用小写命名。

使用new JLabel(imageIcon)或new JButton(imageIcon)在标签或按钮上显示图像图标。程序清单12-8演示如何在标签和按钮上显示图标。这个例子创建两个带图标的标签和两个带图标的按钮，如图12-13所示。

程序清单12-8 TestImageIcon.java

```
1 import javax.swing.*;
2 import java.awt.*;
3
4 public class TestImageIcon extends JFrame {
5   private ImageIcon usIcon = new ImageIcon("image/us.gif");
6   private ImageIcon myIcon = new ImageIcon("image/my.jpg");
7   private ImageIcon frIcon = new ImageIcon("image/fr.gif");
8   private ImageIcon ukIcon = new ImageIcon("image/uk.gif");
```

```
 9
10   public TestImageIcon() {
11     setLayout(new GridLayout(1, 4, 5, 5));
12     add(new JLabel(usIcon));
13     add(new JLabel(myIcon));
14     add(new JButton(frIcon));
15     add(new JButton(ukIcon));
16   }
17
18   /** Main method */
19   public static void main(String[] args) {
20     TestImageIcon frame = new TestImageIcon();
21     frame.setTitle("TestImageIcon");
22     frame.setSize(200, 200);
23     frame.setLocationRelativeTo(null); // Center the frame
24     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
25     frame.setVisible(true);
26   }
27 }
```

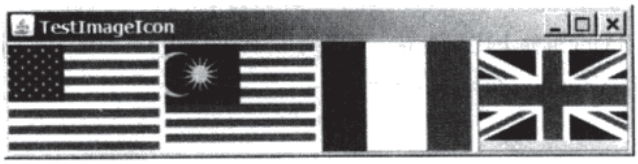


图12-13 在标签和按钮上显示图像图标

注意 GUI组件不能被多个容器共享，因为一个GUI组件只能在一个容器中出现一次。因此，组件和容器之间的关系是用实心菱形表示的组合关系，如图12-1所示。

注意 边界和图标是可以共享的。这样，可以创建一个边界或图标，然后使用它来设置任意一个GUI组件的`border`或`icon`属性。例如，下面的语句就是将两个面板`p1`和`p2`设置为边界`b`：

```
p1.setBorder(b);
p2.setBorder(b);
```

下面的语句是在两个按钮`jbt1`和`jbt2`中设置一个图标：

```
jbt1.setIcon(icon);
jbt2.setIcon(icon);
```

提示 启动画面是应用程序开始启动时显示的图像。如果程序要占用很多时间去加载，就要显示一个启动画面来警示用户。例如，下面的命令：

```
java -splash:image/us.gif TestImageIcon
```

就会实现加载程序`TestImageIcon`时显示一个图像。

关键术语

AWT（抽象窗口工具箱）
heavyweight component（重量级组件）
lightweight component（轻量级组件）
layout manager（布局管理器）
Swing（Swing组件）
splash screen（启动画面）

本章小结

- 每个容器都有一个布局管理器，它按照所需的位置在容器中定位和放置组件。三个简单且常用的布局管理器是`FlowLayout`、`GridLayout`和`BorderLayout`。

- 可以将JPanel作为子容器来将组件分组以得到所需的布局。
- 使用add方法将组件放到JFrame和JPanel。默认情况下，框架的布局是BorderLayout，而JPanel的布局是FlowLayout。
- 可以使用java.awt.Color类设置GUI组件的颜色。颜色是由红、绿和蓝三原色组成的，每种颜色都是用一个无符号的字节值表示它的深度，它的取值范围从0（最暗度）到255（最亮度）。这就是通常所说的RGB模式。
- 为了创建一个Color对象，应该使用new Color(r,g,b)，这里的r、g和b表示该颜色的红色、绿色和蓝色的成分。还可以使用在java.awt.Color中定义的13种标准色（BLACK黑色、BLUE蓝色、CYAN青色、DARK_GRAY深灰、GRAY灰色、GREEN绿色、LIGHT_GRAY淡灰、MAGENTA洋红、ORANGE橘色、PINK粉红、RED大红、WHITE白色和YELLOW黄色）。
- 每个Swing GUI组件都是javax.swing.JComponent的子类，而JComponent则是java.awt.Component的子类。Component中的属性font、background、foreground、height、width和preferredSize都被它们的子类继承，JComponent中的toolTipText和border属性也是如此。
- 可以在任何一个Swing组件上使用边界。可以使用ImageIcon类创建一个图像图标，然后将它显示在标签和按钮上。按钮和边界是可以共享的。

复习题

12.3～12.4节

12.1 哪个类是Java GUI组件类的根？容器类是Component的子类吗？哪个类是Swing GUI组件的根？

12.2 解释像java.awt.Button这样的AWT GUI组件和像javax.swing.JButton这样的Swing GUI组件的不同。

12.3 如何创建一个框架？如何设置框架的大小？如何获取框架的大小？怎样给框架中添加组件？如果将程序清单12-2中的语句frame.setSize(400,300)和语句frame.setVisible(true)互换位置，将会发生什么？

12.4 判断下面语句是真还是假：

（1）可以给框架添加一个按钮。

（2）可以给面板添加一个框架。

（3）可以给框架添加一个面板。

（4）可以给面板或框架添加任意数量的组件。

（5）可以从JButton、JPanel或者JFrame派生一个类。

12.5 下面的程序是要在框架中显示一个按钮，但是什么也没有显示出来。这个程序有什么问题？

```
1 public class Test extends javax.swing.JFrame {
2   public Test() {
3     add(new javax.swing.JButton("OK"));
4   }
5
6   public static void main(String[] args) {
7     javax.swing.JFrame frame = new javax.swing.JFrame();
8     frame.setSize(100, 200);
9     frame.setVisible(true);
10  }
11 }
```

12.6 下面的哪条语句有语法错误？

```
Component c1 = new Component();
JComponent c2 = new JComponent();
Component c3 = new JButton();
```

```
JComponent c4 = new JButton();
Container c5 = new JButton();
c5.add(c4);
Object c6 = new JButton();
c5.add(c6);
```

12.5节

12.7 为什么需要使用布局管理器？框架的默认布局管理器是什么？如何将一个组件添加到框架中？

12.8 描述FlowLayout。如何创建一个FlowLayout管理器？如何将一个组件添加到FlowLayout容器中？添加到FlowLayout容器中的组件数量有限制吗？

12.9 描述GridLayout。如何创建一个GridLayout管理器？如何将一个组件添加到GridLayout容器中？添加到GridLayout容器中的组件数量有限制吗？

12.10 描述BorderLayout。如何创建一个BorderLayout管理器？如何将一个组件添加到BorderLayout容器中？

12.6节

12.11 如何创建一个带特定布局管理器的面板？

12.12 JPanel的默认布局管理器是什么？如何向JPanel添加一个组件？

12.13 可以在面板中使用setTitle方法吗？使用面板的目的是什么？

12.14 由于像JButton这样的GUI组件类是Container的子类，那么是否能将组件添加到按钮中？

12.7～12.8节

12.15 如何创建一种颜色？使用new Color(400,200,300)创建Color时会有什么问题？下面两种颜色哪种比较深：new Color(10,0,0)、new Color(200,0,0)？

12.16 如何创建一种字体？如何找出系统中所有可用的字体？

12.9～12.10节

12.17 如何设置Swing GUI组件的背景色、前景色、字体和工具提示文本？为什么在下面的代码中没有显示工具提示文本？

```
1 import javax.swing.*;
2
3 public class Test extends JFrame {
4   private JButton jbtOK = new JButton("OK");
5
6   public static void main(String[] args) {
7     // Create a frame and set its properties
8     JFrame frame = new Test();
9     frame.setTitle("Logic Error");
10    frame.setSize(200, 100);
11    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
12    frame.setVisible(true);
13  }
14
15  public Test() {
16    jbtOK.setToolTipText("This is a button");
17    add(new JButton("OK"));
18  }
19 }
```

12.18 显示下面代码的输出结果：

```
import javax.swing.*;

public class Test {
  public static void main(String[] args) {
    JButton jbtOK = new JButton("OK");
    System.out.println(jbtOK.isVisible());

    JFrame frame = new JFrame();
```

```
    System.out.println(frame.isVisible());
  }
}
```

12.19 如何从类目录下的image/us.gif文件创建一个ImageIcon。

12.20 如果要将一个按钮如下所示地多次添加到容器中会发生什么情况？它是否会引起语法错误？它是否会引起运行错误？

```
JButton jbt = new JButton();
JPanel panel = new JPanel();
panel.add(jbt);
panel.add(jbt);
panel.add(jbt);
```

12.21 下面的代码会显示三个按钮吗？按钮会显示同样的图标吗？

```
1 import javax.swing.*;
2 import java.awt.*;
3
4 public class Test extends JFrame  {
5   public static void main(String[] args) {
6     // Create a frame and set its properties
7     JFrame frame = new Test();
8     frame.setTitle("ButtonIcons");
9     frame.setSize(200, 100);
10    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
11    frame.setVisible(true);
12  }
13
14  public Test() {
15    ImageIcon usIcon = new ImageIcon("image/us.gif");
16    JButton jbt1 = new JButton(usIcon);
17    JButton jbt2 = new JButton(usIcon);
18
19    JPanel p1 = new JPanel();
20    p1.add(jbt1);
21
22    JPanel p2 = new JPanel();
23    p2.add(jbt2);
24
25    JPanel p3 = new JPanel();
26    p2.add(jbt1);
27
28    add(p1, BorderLayout.NORTH);
29    add(p2, BorderLayout.SOUTH);
30    add(p3, BorderLayout.CENTER);
31  }
32 }
```

12.22 GUI组件可以共享边框或图标吗？

编程练习题

12.5～12.6节

12.1 （使用FlowLayout管理器）编写一个满足下面需求的程序（参见图12-14）：

- 创建一个框架，并且将它的布局设置为FlowLayout。
- 创建两个面板，然后将它们添加到这个框架。
- 每个面板包含三个按钮。面板使用FlowLayout布局管理器。

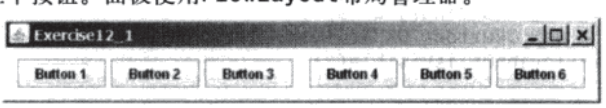


图12-14 练习题12.1将前三个按钮放到一个面板上而将另外三个按钮放到另一个面板上

12.2 （使用BorderLayout管理器）改写前面的程序，创建同样的用户界面，但是在框架上不使用FlowLayout，而是采用BorderLayout。将一个面板放在框架的南区，而将另一个放在中央。

12.3 （使用GridLayout管理器）改写前面的程序，创建同样的用户界面。在这些面板中不用FlowLayout，而是采用两行三列的GridLayout。

12.4 （使用JPanel对按钮分组）改写前面的程序，创建同样的用户界面。不是分别创建面板和按钮，而是定义一个类扩展JPanel类。在面板类中放置三个按钮，然后从这个用户定义的面板类创建两个面板。

12.5 （显示标签）编写一个程序，实现在四个标签上显示四行文本，如图12-15a所示。在每个标签上添加一条线边界。

a) b) c)

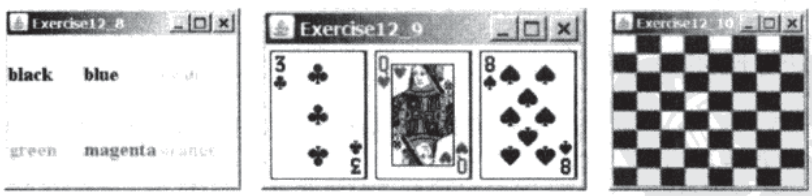
图12-15 a）练习题12.5显示四个标签；b）练习题12.6显示四个图标；c）练习题12.7显示图像图标在标签中的井字游戏的棋盘

12.7～12.10节

12.6 （显示图标）编写一个程序，实现在四个标签上显示四个图标，如图12-15b所示。在每个标签上添加一个线边界（使用你选择的任意图像或者从本书中源代码中获得的任意图像）。

**12.7 （游戏：显示井字游戏的棋盘）显示一个包含九个标签的框架。标签可以显示为一个X字形的图像图标、一个O字形的图像图标，或者什么都不显示，如图12-15c所示。显示什么是随机决定的。使用Math.random()方法产生一个整数0、1或2，相应地显示十字形图像图标、O形图像图标或者什么都不显示。十字形的图像和O形的图像放在www.cs.armstrong.edu/liang/intro8e/book.zip中的图像目录下的x.gif和o.gif文件中。

*12.8 （Swing通用特性）显示包含六个标签的框架。将标签背景色设置为白色。将标签前景色分别设置为黑色、蓝色、青色、绿色、洋红色和橙色，如图12-16a所示。设置每个标签的边界为黄色的线边界。设置每个标签的字体为TimesRoman、加粗、20像素。将每个标签的文本和工具提示文本都设置为它的前景色的名字。



a) b) c)

图12-16 a）框架中放置六个标签；b）随机选择三张牌；c）使用按钮显示一个棋盘

*12.9 （游戏：显示三张牌）显示包含三个标签的框架。每个标签显示一张牌，如图12-16b所示。牌的图像文件命名为1.png，2.png，…，54.png，并且存储在image/card目录中。这三张牌是不同的并且是随机选取的。这些图像文件都可以从www.cs.armstrong.edu/liang/intro8e/book.zip上获取。

*12.10 （游戏：显示一个棋盘）编写一个程序，显示一个棋盘，棋盘中的每一个白色格和黑色格都是将背景色设置为黑色或者白色的JButton，如图12-16c所示。