第18章
Introduction to Java Programming, 8E

applet和多媒体

学习目标

- 将GUI应用程序转换为applet（18.2节）。
- 在网页中嵌入applet（18.3节）。
- 在Web浏览器及applet浏览器中运行applet（18.3.1～18.3.2节）。
- 理解applet安全沙盒模型（18.4节）。
- 编写既可以作为应用程序运行也可以作为applet运行的Java程序（18.5节）。
- 覆盖applet生命周期方法`init`、`start`、`stop`和`destroy`（18.6节）。
- 从HTML中向applet传递字符串参数（18.7节）。
- 开发一个弹跳小球的动画（18.8节）。
- 开发一个井字游戏的applet（18.9节）。
- 使用`URL`类来定位资源（图像和音频）（18.10节）。
- 在Java程序中播放音频（18.11节）。

18.1 引言

浏览网页时，经常会看到使用Java开发的图形用户界面和动画。这些程序称作Java applet。假设希望开发一个九宫格游戏的Java applet，如图18-1所示，该如何编写这个程序呢？

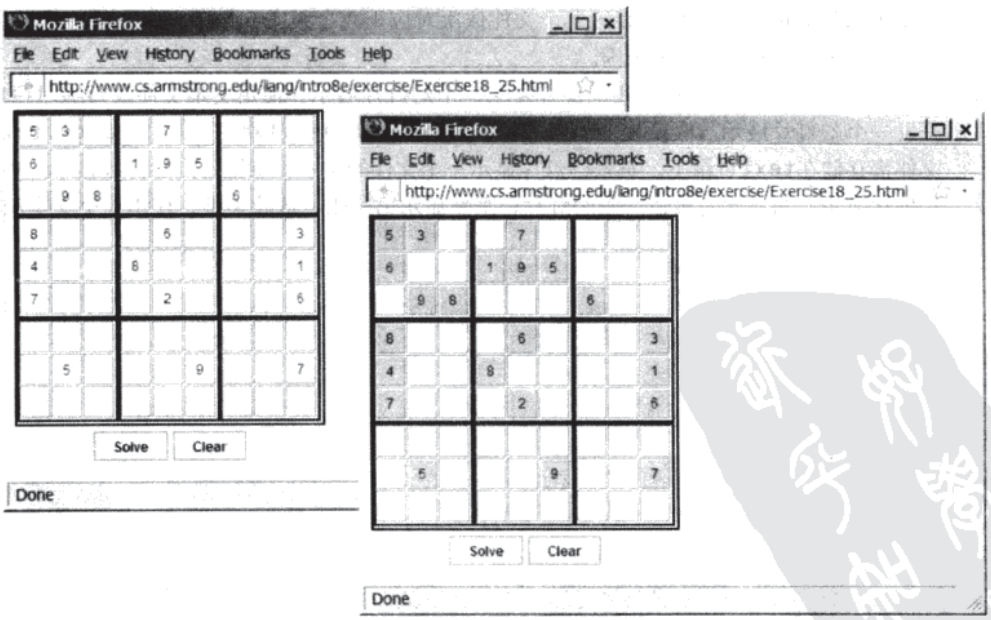


图18-1 在Web浏览器上显示九宫格游戏

在本章中，将学会如何编写Java applet，考察applet和Web浏览器之间的关系，探讨应用程序和applet之间的异同点，还将学习如何创建带图像和音频的多媒体Java应用程序和Java applet。

18.2 开发applet

到目前为止，所使用的只是Java应用程序。但是，所学到的关于如何编写应用程序的知识都可以用来编写applet。应用程序和applet共享许多通用的程序设计特性，尽管它们在某些方面有些小的不同。例如，每个应用程序都必须有main方法，该方法被Java解释器调用。另一方面，Java applet是不需要main方法的，它们可以在Web浏览器环境下运行。因为applet是从Web页面调用的，所以Java提供了能让applet在Web浏览器中运行的特殊特性。

Applet类提供了一个基本的框架结构，使得applet可以在Web浏览器中运行。每个Java应用程序都有一个在应用程序开始时执行的main方法，而applet没有main方法，它们依靠浏览器来运行。每个applet都是java.applet.Applet的子类。Applet类是一个AWT类，它不能和Swing组件一起工作。要在Java applet中使用Swing组件，需要通过扩展javax.swing.JApplet来创建一个Java applet，这里的javax.swing.JApplet是java.applet.Applet的一个子类。

开发的每一个Java GUI程序都可以通过将JFrame替换为JApplet同时删除main方法转换为一个applet。图18-2a显示了一个Java GUI应用程序，它可以转换为一个如图18-2b所示的Java applet。

```
import javax.swing.*;

public class DisplayLabel extends JFrame {
  public DisplayLabel() {
    add(new JLabel("Great!", JLabel.CENTER));
  }

  public static void main(String[] args) {
    JFrame frame = new DisplayLabel();
    frame.setTitle("DisplayLabel");
    frame.setSize(200, 100);
    frame.setLocationRelativeTo(null);
    frame.setDefaultCloseOperation(
      JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);
  }
}
```

a) GUI应用程序

```
import javax.swing.*;
                                    JApplet
public class DisplayLabel extends JFrame {
  public DisplayLabel() {
    add(new JLabel("Great!", JLabel.CENTER));
  }

  public static void main(String[] args) {
    JFrame frame = new DisplayLabel();
    frame.setTitle("DisplayLabel");
    frame.setSize(200, 100);
    frame.setLocationRelativeTo(null);
    frame.setDefaultCloseOperation(
      JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);
  }
}
```

b) applet

图18-2 可以将一个GUI应用程序转换为一个applet

程序清单18-1给出这个applet的完整代码。

程序清单18-1 DisplayLabel.java

```
1 import javax.swing.*;
2
3 public class DisplayLabel extends JApplet {
4   public DisplayLabel() {
5     add(new JLabel("Great!", JLabel.CENTER));
6   }
7 }
```

就像JFrame一样，JApplet是一个可以包含其他GUI组件的容器（参见图14-1中的GUI类图）。

18.3 HTML文件和<applet>标记

为了从浏览器运行一个applet，需要创建带<applet>标记的HTML文件。

HTML是一种在网页上展示静态文档的标识语言。HTML使用标记指示Web浏览器如何绘制Web页面，HTML包含一个称为<applet>的标记，该标记将applet嵌入Web页面。

程序清单18-2中的HTML文件调用DisplayLabel.class：

程序清单18-2 DisplayLabel.html

```
<html>
  <head>
    <title>Java Applet Demo</title>
  </head>
  <body>
    <applet
      code = "DisplayLabel.class"
      width = 250
      height = 50>
    </applet>
  </body>
</html>
```

对Web浏览器来说，标记（tag）是一种指令。浏览器会解释标记，然后决定如何显示这个HTML文档中后续的内容，或者决定如何处理后续的内容。这些标记用尖括号括起来。标记中的第一个单词称为标记名（tag name），它描述标记的功能。标记还可以有一些附加的属性，有时候在等号后会有一个值，它会进一步定义标记的行为。例如，在前面的HTML文件中，<applet>是标记名，而code、width和height都是属性。属性width和height表明applet的矩形视图区域的属性。

大多数标记都有一个开始标记（start tag）和对应的结束标记（end tag）。标记对开始标记和结束标记之间的内容会产生特定的效果。例如，<applet...>...</applet>会告诉浏览器显示一个applet。结束标记总是在开始标记的名字之前加一条斜杠。

HTML文档以标记<html>开始，声明这个文档是用HTML语言编写的。每个文档都有两部分：头（head）和体（body），分别用<head>标记和<body>标记来定义。头部分包含包括在<title>中的文档标题，以及绘制文档时浏览器可能使用的其他信息，而体部分包含文档的实际内容。头是可选的。

<applet>标记的完整语法如下：

```
<applet
  [codebase = applet_url]
  code = classfilename.class
  width = applet_viewing_width_in_pixels
  height = applet_viewing_height_in_pixels
  [archive = archivefile]
  [vspace = vertical_margin]
  [hspace = horizontal_margin]
  [align = applet_alignment]
  [alt = alternative_text]
>
<param name = param_name1 value = param_value1>
<param name = param_name2 value = param_value2>
...
<param name = param_name3 value = param_value3>
</applet>
```

属性code、width和height是必需的，其余属性是可选的。<param>标记将在18.7节中介绍，其他属性解释如下：

1）codebase指定类该加载到哪里。如果不使用这个属性，Web浏览器会从HTML页面所在的目录加载applet。如果applet与HTML网页不在同一目录下，则必须为浏览器指定装载applet的applet_url。利用这一属性，可以从Internet的任何地方加载类。必要时，applet动态地加载所使用的类。

2）archive指示浏览器加载一个存档文件，该文件包含运行applet所需要的所有类文件。存档允许Web浏览器从一个压缩文件一次性加载所有的类，这样可以减少加载时间同时提高性能。要想建立存档文件，请参见补充材料III.Q。

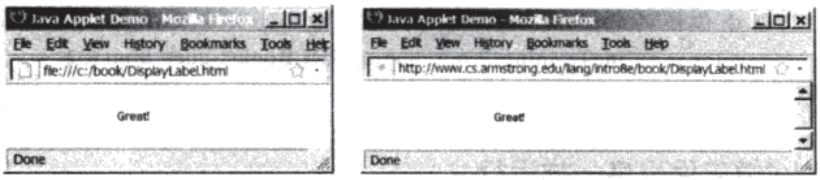
3）vspace和hspace指定applet周围的垂直方向和水平方向空白边界的大小，它们的单位是像素。

4）align指定applet在浏览器中是如何对齐的。使用下面的九个值之一：left、right、top、texttop、middle、absmiddle、baseline、bottom或absbottom。

5）alt指定浏览器不能运行Java时显示的文本。

18.3.1 从Web浏览器查看applet

要从一个Web浏览器中显示applet，打开applet的HTML文件（例如，DisplayLabel.html）即可。它的输出结果如图18-3a所示。



a） b）

图18-3 a）中的DisplayLabel程序是从本地主机加载的，而b）中的DisplayLabel是从Web服务器加载的

要想在网上可访问applet，需要将DisplayLabel.class和DisplayLabel.html都存在一个Web服务器上，如图18-4所示。可以从一个恰当的URL中浏览一个applet。例如，可以向Web服务器www.cs.armstrong.edu/上传这两个文件。如图18-3b所示，可以从www.cs.armstrong.edu/liang/intro8e/book/DisplayLabel.html访问这个applet。

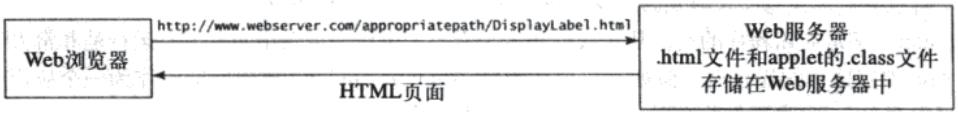
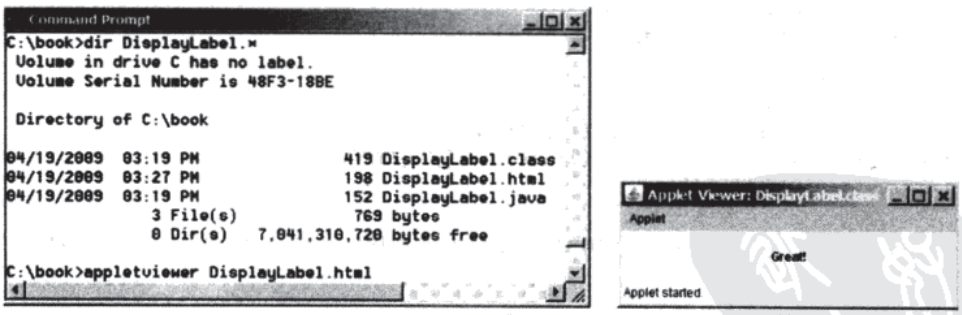


图18-4 Web浏览器请求来自Web服务器的一个HTML网页

18.3.2 使用applet查看器工具查看applet

可以使用applet查看器工具测试applet。在DOS命令行中，从目录c:\book中使用appletviewer命令就可以调用这个工具，如图18-5a所示。它的输出结果如图18-5b所示。



a） b）

图18-5 使用appletviewer命令在applet查看器工具中运行一个Java applet

applet查看器工具的功能就相当于一个Web浏览器。在没有启动Web浏览器的开发过程中，这是测试applet的一个很简便的方法。

18.4 applet安全限制

Java使用所谓的“沙盒安全模型”来执行applet，以防止毁灭性的程序危害到运行浏览器的系统。不允许applet使用“沙盒”之外的资源。沙盒会特别地限制下面的动作：

- 不允许applet对计算机的文件系统进行读取或写入。否则，它们会损坏这些文件，还会传播病毒。
- 不允许applet运行浏览器所在的计算机上的任何程序。否则，它们可能会调用有破坏力的本地程序，并且破坏用户计算机的本地系统。
- 除了在服务器上存储applet之外，不允许applet同用户的计算机和其他任何计算机之间建立任何连接。这个限制是为了防止在用户不知道的情况下，applet将用户的计算机连接到另一台计算机。

注意 可以创建信任的applet（signed applet）规避这些安全限制。可以参见网页java.sun.com/developer/onlineTraining/Programming/JDCBook/signed.html获取如何创建信任applet的详细指示。

18.5 让applet像应用程序一样运行

尽管JFrame类和JApplet类有一些差别，但是它们有很多共同之处。由于它们都是Container类的子类，所以，它们所有的用户界面组件、布局管理器以及事件处理特征都是一样的。但是，应用程序是Java解释器从静态main方法调用的，而applet是由Web浏览器运行的。Web浏览器使用applet的无参数构造方法创建applet的一个实例，然后控制和执行这个applet。

一般来说，applet都能转换为应用程序而不损失任何功能。只要不违反applet上的强制性安全限制，应用程序也可以转换为applet。可以在applet中实现一个main方法，这样，applet就能像应用程序一样运行。这个特点既有理论意义又有实践意义。理论上讲，它模糊了applet和应用程序之间的区别。可以编写一个既是applet又是应用程序的类。从实践的角度来看，一个程序可以用两种方式运行是非常方便的。

该如何编写这样的程序呢？假如有一个名为MyApplet的applet，要使它能够作为一个应用程序来运行，就只需要在applet中添加并实现一个main方法，如下所示：

```
public static void main(String[] args) {
  // Create a frame
  JFrame frame = new JFrame("Applet is in the frame");

  // Create an instance of the applet
  MyApplet applet = new MyApplet();

  // Add the applet to the frame
  frame.add(applet, BorderLayout.CENTER);

  // Display the frame
  frame.setSize(300, 300);
  frame.setLocationRelativeTo(null); // Center the frame
  frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  frame.setVisible(true);
}
```

可以通过在程序清单18-3中添加一个main方法来修改程序清单18-1中的DisplayLabel类，以使它可以独立运行。

程序清单18-3 有main方法的新DisplayLabel.java

```
 1 import javax.swing.*;
 2
 3 public class DisplayLabel extends JApplet {
 4   public DisplayLabel() {
 5     add(new JLabel("Great!", JLabel.CENTER));
 6   }
 7
 8   public static void main(String[] args) {
 9     // Create a frame
10     JFrame frame = new JFrame("Applet is in the frame");
11
12     // Create an instance of the applet
13     DisplayLabel applet = new DisplayLabel();
```

```
14
15      // Add the applet to the frame
16      frame.add(applet);
17
18      // Display the frame
19      frame.setSize(300, 100);
20      frame.setLocationRelativeTo(null); // Center the frame
21      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
22      frame.setVisible(true);
23    }
24 }
```

当从Web浏览器运行applet时，浏览器会创建一个applet的实例，然后显示它。当独立运行applet时，就会调用main方法来创建一个存放applet的框架（第10行）。创建applet（第13行）并且将它添加到这个框架中（第16行）。这个框架在第22行显示。

18.6 applet生命周期方法

实际上，applet是从applet容器（applet container）运行的，这个容器是Web浏览器的一个插件。类Applet包括init()、start()、stop()和destroy()方法，这些方法都称为生命周期方法（life-cycle method）。这些方法都被applet容器调用以控制一个applet的执行。它们在Applet类中用空体来实现。所以，默认情况下，它们什么都不做。可以在Applet的子类中覆盖它们以完成所需的操作。图18-6显示applet容器是如何调用这些方法的。

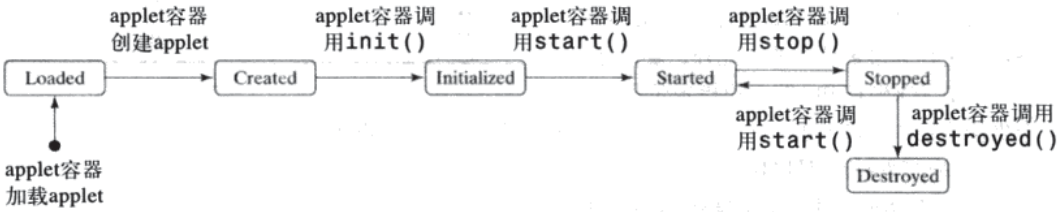


图18-6 applet容器使用init、start、stop和destroy方法来控制applet

18.6.1 init方法

创建applet之后就会调用init方法。如果Applet的子类有一个初始化过程要完成，那么该子类就应覆盖init方法。通常，这个方法实现的功能包括从HTML网页的<applet>标记中获取字符串参数值。

18.6.2 start方法

在init方法之后就会调用start方法。当用户浏览过其他页面之后返回到包含这个applet的Web页面时，该方法也会被调用。

每当访问包含applet的Web网页时，如果还有任何需要完成的操作，Applet的子类就会覆盖这个方法。例如，一个带动画的applet可以启动定时器来重新开始动画。

18.6.3 stop方法

stop方法与start方法恰好相反，start方法是在用户返回包含applet的网页时调用的，而stop方法是在用户离开这个网页时调用的。

每当包含applet的Web页面不再可见时，如果还有其他需要执行的操作，Applet的子类就会覆盖这个方法。例如，一个带动画的applet可能会停止定时器来暂停动画。

18.6.4 destroy方法

当浏览器正常退出时就会调用destroy方法，以通知applet不再需要它并且应该释放它所占有的资

源。stop方法总是在destroy方法之前调用。

如果Applet子类在销毁之前还有需要完成的操作，这个Applet的子类就会覆盖destroy方法。通常情况下是不需要覆盖这个方法的，除非希望释放创建applet所占有的特定资源。

18.7 给applet传递字符串

在9.5节中，已经学习了如何从命令行向Java应用程序传递字符串。字符串是作为字符串数组传递给main方法的。当应用程序开始时，main方法就可以使用这些字符串。但是，在applet中没有main方法，所以不能通过Java解释器从命令行运行。

那么，applet是如何接收参数的呢？在本节中，将学习如何向Java applet传递字符串。要将参数传递给applet，必须在HTML文件中声明它，而且必须在applet初始化时由applet读取。参数是使用<param>标记声明的。<param>标记必须嵌入到<applet>标记中且没有对应的结束标记。它的语法如下所示：

```
<param name = parametername value = stringvalue />
```

这个标记表明一个参数和它对应的字符串值。

注意 在HTML代码中，参数名与参数值之间没有逗号分隔。HTML的参数名是不区分大小写的。

表18-1 DisplayMessage applet的参数名和参数值

参数名	参数值
MESSAGE	"Welcome to Java"
X	20
Y	30

假设要编写一个applet来显示一条消息。这条消息是作为参数传递的。此外，要让这条消息显示在有x坐标和y坐标的指定位置，这两个坐标值也作为参数进行传递。表18-1中列出了参数以及它们的值。

程序清单18-4给出HTML源文件。

程序清单18-4 DisplayMessage.html

```
<html>
  <head>
    <title>Passing Strings to Java Applets</title>
  </head>
  <body>
    <p>This applet gets a message from the HTML
       page and displays it.</p>
    <applet
      code = "DisplayMessage.class"
      width = 200
      height = 50
      alt = "You must have a Java 2-enabled browser to view the applet"
    >
      <param name = MESSAGE value = "Welcome to Java" />
      <param name = X value = 20 />
      <param name = Y value = 30 />
    </applet>
  </body>
</html>
```

为了从applet中读取参数，使用在Applet类中定义的下述方法：

```
public String getParameter(String parametername);
```

它会返回指定参数的值。

程序清单18-5给出这个applet。这个applet的运行示例如图18-7所示。

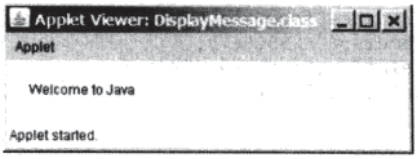


图18-7 这个applet显示从HTML页面传递来的消息Welcome to Java

程序清单18-5 DisplayMessage.java

```
1 import javax.swing.*;
2
3 public class DisplayMessage extends JApplet {
4   /** Initialize the applet */
5   public void init() {
6     // Get parameter values from the HTML file
7     String message = getParameter("MESSAGE");
8     int x = Integer.parseInt(getParameter("X"));
9     int y = Integer.parseInt(getParameter("Y"));
10
11    // Create a message panel
12    MessagePanel messagePanel = new MessagePanel(message);
13    messagePanel.setXCoordinate(x);
14    messagePanel.setYCoordinate(y);
15
16    // Add the message panel to the applet
17    add(messagePanel);
18  }
19 }
```

这个程序在init方法中获取来自HTML文件的参数值。这些值是使用getParameter方法获取的字符串（第7～9行）。因为x和y都是int型的，所以，程序使用Integer.parseInt(string)方法将一个数值字符串转化为一个int值。

如果将HTML文件中的Welcome to Java改为Welcome to HTML，然后在Web浏览器上重新加载这个HTML文件，就会看到显示Welcome to HTML。同样，也可以改变x和y的值，在所需的位置显示这条信息。

警告 Applet类的getParameter方法只能在创建一个applet的实例之后调用。因此，这个方法不能在applet类的构造方法中调用。应该从init方法中调用它。

可以添加一个main方法，让这个applet能够独立运行。当这个applet作为一个applet运行时会从HTML文件中读取参数，而当它独立运行时会从命令行读取参数。除了有一个新的main方法以及一个表明这个程序是作为一个applet还是一个应用程序来运行的名为isStandalone的变量值之外，程序清单18-6所示的程序和DisplayMessage是一样的。

程序清单18-6 DisplayMessageApp.java

```
1 import javax.swing.*;
2 import java.awt.Font;
3 import java.awt.BorderLayout;
4
5 public class DisplayMessageApp extends JApplet {
6   private String message = "A default message"; // Message to display
7   private int x = 20; // Default x-coordinate
8   private int y = 20; // Default y-coordinate
9
10  /** Determine whether it is an application */
11  private boolean isStandalone = false;
12
13  /** Initialize the applet */
14  public void init() {
15    if (!isStandalone) {
```

```
16        // Get parameter values from the HTML file
17        message = getParameter("MESSAGE");
18        x = Integer.parseInt(getParameter("X"));
19        y = Integer.parseInt(getParameter("Y"));
20      }
21
22      // Create a message panel
23      MessagePanel messagePanel = new MessagePanel(message);
24      messagePanel.setFont(new Font("SansSerif", Font.BOLD, 20));
25      messagePanel.setXCoordinate(x);
26      messagePanel.setYCoordinate(y);
27
28      // Add the message panel to the applet
29      add(messagePanel);
30    }
31
32    /** Main method to display a message
33       @param args[0] x-coordinate
34       @param args[1] y-coordinate
35       @param args[2] message
36       */
37    public static void main(String[] args) {
38      // Create a frame
39      JFrame frame = new JFrame("DisplayMessageApp");
40
41      // Create an instance of the applet
42      DisplayMessageApp applet = new DisplayMessageApp();
43
44      // It runs as an application
45      applet.isStandalone = true;
46
47      // Get parameters from the command line
48      applet.getCommandLineParameters(args);
49
50      // Add the applet instance to the frame
51      frame.add(applet, BorderLayout.CENTER);
52
53      // Invoke applet's init method
54      applet.init();
55      applet.start();
56
57      // Display the frame
58      frame.setSize(300, 300);
59      frame.setLocationRelativeTo(null); // Center the frame
60      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
61      frame.setVisible(true);
62    }
63
64    /** Get command-line parameters */
65    private void getCommandLineParameters(String[] args) {
66      // Check usage and get x, y and message
67      if (args.length != 3) {
68        System.out.println(
69          "Usage: java DisplayMessageApp x y message");
70        System.exit(0);
71      }
72      else {
73        x = Integer.parseInt(args[0]);
74        y = Integer.parseInt(args[1]);
75        message = args[2];
76      }
77    }
78  }
```

将这个程序作为一个applet运行时，忽略main方法。若将这个程序作为应用程序运行，调用main方

法。这个程序作为应用程序和作为applet的运行示例如图18-8所示。

图18-8 DisplayMessageApp类既可以作为一个应用程序运行，也可以作为一个applet运行

main方法创建了一个JFrame的对象frame，还创建了一个JApplet的对象applet，然后，将这个applet放到框架frame中，接着调用它的init方法，应用程序就会像applet一样运行。

main方法将isStandalone设置为true（第45行），所以不要在调用init方法时尝试重新获取HTML参数。

当组件添加到applet之后，就会调用setVisible(true)方法（第61行），然后将applet添加到框架中以确保这个组件是可见的。否则，当框架开始时不显示组件。

重要的教学注意 从现在开始，所有的GUI示例都将创建为带main方法的applet。这样，既可以将程序当作apple运行，也可以当作应用程序运行。为了简洁，文中不列出main方法。

18.8 实例学习：弹跳的小球

本节给出一个applet，显示一个小球在面板上弹跳。使用两个按钮暂停和重新开始小球的运动，使用一个滚动条控制小球弹跳的速度，如图18-9所示。

图18-9 用Suspend按钮、Resume按钮以及滚动条控制小球的运动

下面是完成这个例子的主要步骤：

1）创建一个JPanel的子类，将它命名为Ball，用来显示小球的弹跳，如程序清单18-7所示。

2）创建一个JPanel的子类，将它命名为BallControl，包含一个小球、一个滚动条、两个控制按钮Suspend和Resume，如程序清单18-8所示。

3）创建一个名为BounceBallApp的applet，它包含BallControl的一个实例，使得applet能够独立运行，如程序清单18-9所示。

这些类之间的关系如图18-10所示。

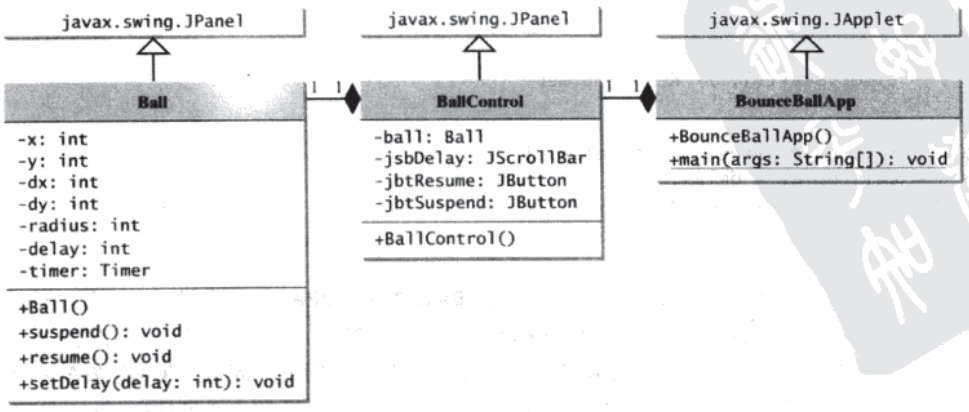


图18-10 BounceBallApp包含BallControl，而BallControl包含Ball

程序清单18-7 **Ball.java**

```
1 import javax.swing.Timer;
2 import java.awt.*;
3 import javax.swing.*;
4 import java.awt.event.*;
5
6 public class Ball extends JPanel {
7   private int delay = 10;
8
9   // Create a timer with delay 1000 ms
10  private Timer timer = new Timer(delay, new TimerListener());
11
12  private int x = 0; private int y = 0; // Current ball position
13  private int radius = 5; // Ball radius
14  private int dx = 2; // Increment on ball's x-coordinate
15  private int dy = 2; // Increment on ball's y-coordinate
16
17  public Ball() {
18    timer.start();
19  }
20
21  private class TimerListener implements ActionListener {
22    /** Handle the action event */
23    public void actionPerformed(ActionEvent e) {
24      repaint();
25    }
26  }
27
28  protected void paintComponent(Graphics g) {
29    super.paintComponent(g);
30
31    g.setColor(Color.red);
32
33    // Check boundaries
34    if (x < radius) dx = Math.abs(dx);
35    if (x > getWidth() - radius) dx = -Math.abs(dx);
36    if (y < radius) dy = Math.abs(dy);
37    if (y > getHeight() - radius) dy = -Math.abs(dy);
38
39    // Adjust ball position
40    x += dx;
41    y += dy;
42    g.fillOval(x - radius, y - radius, radius * 2, radius * 2);
43  }
44
45  public void suspend() {
46    timer.stop(); // Suspend timer
47  }
48
49  public void resume() {
50    timer.start(); // Resume timer
51  }
52
53  public void setDelay(int delay) {
54    this.delay = delay;
55    timer.setDelay(delay);
56  }
57 }
```

在16.12节中介绍了如何使用Timer来控制动画。Ball类扩展JPanel类来显示一个移动的球。定时器监听器实现ActionListener来监听ActionEvent事件（第21行）。第10行为Ball创建一个Timer对象。当构造一个Ball的实例时，就会启动第18行的定时器。这个定时器以固定的频率触发ActionEvent事件。监听器在第24行做出响应重画小球，以得到小球运动的动画。球的中心坐标在（x，y）处，它在

下一次显示时就会变为（x+dx，y+dy）（第40～41行）。suspend方法和resume方法（第45～51行）可以用来停止和启动定时器。setDelay(int)方法（第53～56行）设置一个新延时。

程序清单18-8 BallControl.java

```
1 import javax.swing.*;
2 import java.awt.event.*;
3 import java.awt.*;
4
5 public class BallControl extends JPanel {
6   private Ball ball = new Ball();
7   private JButton jbtSuspend = new JButton("Suspend");
8   private JButton jbtResume = new JButton("Resume");
9   private JScrollBar jsbDelay = new JScrollBar();
10
11  public BallControl() {
12    // Group buttons in a panel
13    JPanel panel = new JPanel();
14    panel.add(jbtSuspend);
15    panel.add(jbtResume);
16
17    // Add ball and buttons to the panel
18    ball.setBorder(new javax.swing.border.LineBorder(Color.red));
19    jsbDelay.setOrientation(JScrollBar.HORIZONTAL);
20    ball.setDelay(jsbDelay.getMaximum());
21    setLayout(new BorderLayout());
22    add(jsbDelay, BorderLayout.NORTH);
23    add(ball, BorderLayout.CENTER);
24    add(panel, BorderLayout.SOUTH);
25
26    // Register listeners
27    jbtSuspend.addActionListener(new ActionListener() {
28      public void actionPerformed(ActionEvent e) {
29        ball.suspend();
30      }
31    });
32    jbtResume.addActionListener(new ActionListener() {
33      public void actionPerformed(ActionEvent e) {
34        ball.resume();
35      }
36    });
37    jsbDelay.addAdjustmentListener(new AdjustmentListener() {
38      public void adjustmentValueChanged(AdjustmentEvent e) {
39        ball.setDelay(jsbDelay.getMaximum() - e.getValue());
40      }
41    });
42  }
43 }
```

BallControl类扩展JPanel类，显示一个球、一个滚动条和两个控制按钮。当点击Suspend按钮时，就会调用球的suspend()方法使球暂停运动（第29行）。当点击Resume按钮时，就会调用球的resume()方法使球重新开始运动（第34行）。可以使用滚动条改变小球弹跳的速度。

程序清单18-9 BounceBallApp.java

```
1 import java.awt.*;
2 import javax.swing.*;
3
4 public class BounceBallApp extends JApplet {
5   public BounceBallApp() {
6     add(new BallControl());
7   }
8 }
```

BounceBallApp类只在applet中放置了一个BallControl的实例。该applet还提供了main方法（为

了简明，没有在清单中列出来），以便可以独立运行它。

18.9 实例学习：井字游戏

从本章和前面各章的例子中我们已经学习了对象、类、数组、类的继承、GUI、事件驱动程序设计以及applet。现在到了应用所学知识开发一些复杂项目的时候了。本节将开发一个玩流行的井字游戏（TicTacToe）的Java applet。

在井字游戏中，两个玩家在3×3的网格中轮流将各自的标记填在空格中（一个人用X，另一个人用O）。如果一个玩家在网格的水平方向、垂直方向或对角线方向上放了三个连续标记，游戏就以这个玩家得胜而告终。若网格的所有单元格都标满了标记还没有产生优胜者，就会出现平局（没有胜者）。图18-11显示了这个例子的典型运行示例。

a）X玩家赢了游戏　b）平局——没有获胜者　c）O玩家赢了游戏

图18-11 两个玩家玩井字游戏

到现在为止，我们见过的所有例子的动作都很简单，都很容易用类来模拟。但是井字游戏的动作有些复杂。为了创建模拟游戏的行为的类，需要研究和了解这个游戏。

假设开始时所有的单元格都是空的，并且第一个玩家用X标记，第二个玩家用O标记。要在单元格上作标记，玩家应该将鼠标指针放在这个单元格上，然后点击它。如果这个单元格为空，就显示标记（X或O）。如果这个单元格已经被填充，则忽略这个玩家的动作。

从前面的描述可以知道，单元格显然是处理鼠标点击事件和显示标记的GUI对象。这样的对象可以是按钮，也可以是面板。在面板上画图比在按钮上画图具有更高的灵活性，因为在面板上可以画任意大小的标记（X或O），但是，按钮上的标记只能显示为文本标签。因此，应该选用面板建模单元格。怎样才能知道单元格的状态（空、X或O）呢？可以使用单元格类`Cell`中名为`token`的`char`型属性来解决这个问题。`Cell`类负责点击空单元格时画出标记。因此，需要编写代码来监听`MouseEvent`事件，以及绘制标记X和O形状的代码。`Cell`类可以定义为如图18-12所示。

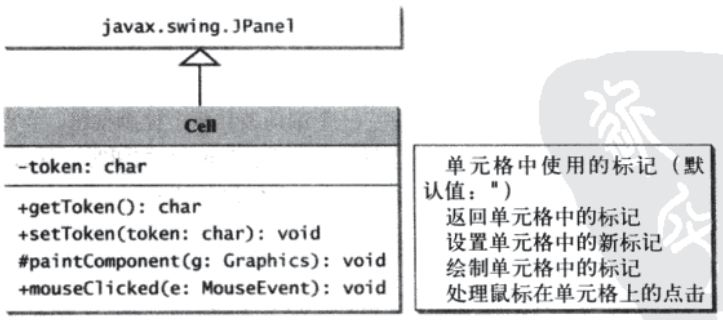


图18-12 `Cell`类在单元格上绘制标记

井字棋盘由9个单元格组成，使用`new Cell[3][3]`创建。为了判断轮到哪个玩家出棋，可以引入名为`whoseTurn`的`char`型变量，该变量的初始值为X，然后变为O，接下来，每当填充新单元格，它就在X和O之间依次转换。当游戏结束时，`whoseTurn`设置为`' '`（空）。

如何才能知道这场游戏是否结束，是否产生了优胜者？如果有优胜者，那么谁是优胜者？可以创建

一个名为isWon(char token)的方法来判断指定标记是否是优胜者，也可以创建一个名为isFull()的方法来判断是否所有的单元格都被占满。

很显然，前面的分析出现了两个类。一个类是处理单个单元格上操作的Cell类；另一个类是玩整个游戏并处理所有单元格的TicTacToe类。这两个类之间的关系如图18-13所示。

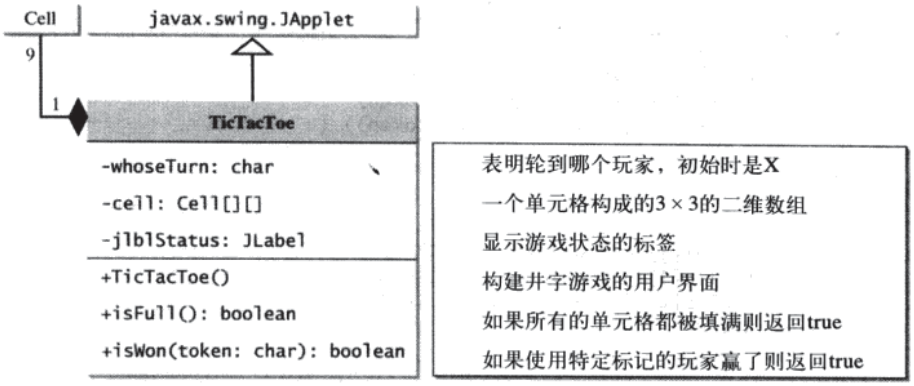


图18-13 TicTacToe类包含9个单元格

因为Cell类只是为了支持TicTacToe类，所以，它可以定义为TicTacToe类的一个内部类。完整的程序在程序清单18-10中给出。

程序清单18-10 TicTacToe.java

```
1 import java.awt.*;
2 import java.awt.event.*;
3 import javax.swing.*;
4 import javax.swing.border.LineBorder;
5
6 public class TicTacToe extends JApplet {
7   // Indicate which player has a turn; initially it is the X player
8   private char whoseTurn = 'X';
9
10  // Create and initialize cells
11  private Cell[][] cells =  new Cell[3][3];
12
13  // Create and initialize a status label
14  private JLabel jlblStatus = new JLabel("X's turn to play");
15
16  /** Initialize UI */
17  public TicTacToe() {
18    // Panel p to hold cells
19    JPanel p = new JPanel(new GridLayout(3, 3, 0, 0));
20    for (int i = 0; i < 3; i++)
21      for (int j = 0; j < 3; j++)
22        p.add(cells[i][j] = new Cell());
23
24    // Set line borders on the cells panel and the status label
25    p.setBorder(new LineBorder(Color.red, 1));
26    jlblStatus.setBorder(new LineBorder(Color.yellow, 1));
27
28    // Place the panel and the label to the applet
29    add(p, BorderLayout.CENTER);
30    add(jlblStatus, BorderLayout.SOUTH);
31  }
32
33  /** Determine whether the cells are all occupied */
34  public boolean isFull() {
35    for (int i = 0; i < 3; i++)
36      for (int j = 0; j < 3; j++)
```

```
37          if (cells[i][j].getToken() == ' ')
38            return false;
39
40        return true;
41      }
42
43      /** Determine whether the player with the specified token wins */
44      public boolean isWon(char token) {
45        for (int i = 0; i < 3; i++)
46          if ((cells[i][0].getToken() == token)
47              && (cells[i][1].getToken() == token)
48              && (cells[i][2].getToken() == token)) {
49            return true;
50          }
51
52        for (int j = 0; j < 3; j++)
53          if ((cells[0][j].getToken() ==  token)
54              && (cells[1][j].getToken() == token)
55              && (cells[2][j].getToken() == token)) {
56            return true;
57          }
58
59        if ((cells[0][0].getToken() == token)
60            && (cells[1][1].getToken() == token)
61            && (cells[2][2].getToken() == token)) {
62          return true;
63        }
64
65        if ((cells[0][2].getToken() == token)
66            && (cells[1][1].getToken() == token)
67            && (cells[2][0].getToken() == token)) {
68          return true;
69        }
70
71        return false;
72      }
73
74      // An inner class for a cell
75      public class Cell extends JPanel {
76        // Token used for this cell
77        private char token = ' ';
78
79        public Cell() {
80          setBorder(new LineBorder(Color.black, 1)); // Set cell's border
81          addMouseListener(new MyMouseListener());  // Register listener
82        }
83
84        /** Return token */
85        public char getToken() {
86          return token;
87        }
88
89        /** Set a new token */
90        public void setToken(char c) {
91          token = c;
92          repaint();
93        }
94
95        /** Paint the cell */
96        protected void paintComponent(Graphics g) {
97          super.paintComponent(g);
98
99          if (token == 'X') {
100           g.drawLine(10, 10, getWidth() - 10, getHeight() - 10);
101           g.drawLine(getWidth() - 10, 10, 10, getHeight() - 10);
```

```
102       }
103       else if (token == 'O') {
104         g.drawOval(10, 10, getWidth() - 20, getHeight() - 20);
105       }
106     }
107
108     private class MyMouseListener extends MouseAdapter {
109       /** Handle mouse click on a cell */
110       public void mouseClicked(MouseEvent e) {
111         // If cell is empty and game is not over
112         if (token == ' ' && whoseTurn != ' ') {
113           setToken(whoseTurn); // Set token in the cell
114
115           // Check game status
116           if (isWon(whoseTurn)) {
117             jlblStatus.setText(whoseTurn + " won! The game is over");
118             whoseTurn = ' '; // Game is over
119           }
120           else if (isFull()) {
121             jlblStatus.setText("Draw! The game is over");
122             whoseTurn = ' '; // Game is over
123           }
124           else {
125             // Change the turn
126             whoseTurn = (whoseTurn == 'X') ? 'O': 'X';
127             // Display whose turn
128             jlblStatus.setText(whoseTurn + "'s turn");
129           }
130         }
131       }
132     }
133   }
134 }
```

TicTacToe类使用放置在GridLayout的面板上的9个单元格来初始化用户界面（第19～22行）。名为jlblStatus的标签用来显示游戏的状态（第14行）。变量whoseTurn（第8行）用来跟踪下一个要放在单元格中的标记的类型。isFull方法（第34～41行）和isWon方法（第44～72行）用来判断这个游戏的状态。

由于Cell类是TicTacToe类中的内部类，所以，可以从类Cell中引用TicTacToe类中定义的变量（whoseTurn）和方法（isFull和isWon）。内部类可以使程序简洁明了。如果没有把Cell类声明为TicTacToe的内部类，为了可以在Cell中使用TicTacToe中的变量和方法，就必须给Cell传递一个TicTacToe对象。编程练习题18.6要求不使用内部类来改写这个程序。

MouseEvent的监听器是为单元格而注册的（第81行）。如果游戏没有结束时点击空单元格，那么在单元格中会设置一个标记（第113行）。如果游戏结束，whoseTurn设置为' '（空）（第118行和第122行）。否则，whoseTurn就会轮换到新的一轮（第126行）。

提示 采用逐步扩充的方法开发和测试这一类Java项目。前面的程序可以分为五个步骤：

1）部署用户界面，然后在单元格中显示一个固定标记X。

2）使单元格能够响应鼠标点击以显示固定标记X。

3）协调两个玩家，以便交替地显示标记X和O。

4）判断是否有玩家获胜，或者所有的单元格都被占满且仍无获胜者。

5）一旦一个玩家移动一步，就在标签上显示一条消息。

18.10 使用URL类定位资源

前面已经使用ImageIcon类从一个图像文件中创建一个图标，并使用setIcon方法或构造方法将图

标放在类似按钮或标签的GUI组件中。例如，下面的语句创建一个ImageIcon，并且将它设置在JLabel对象jlbl上：

```
ImageIcon imageIcon = new ImageIcon("c:\\book\\image\\us.gif");
jlbl.setIcon(imageIcon);
```

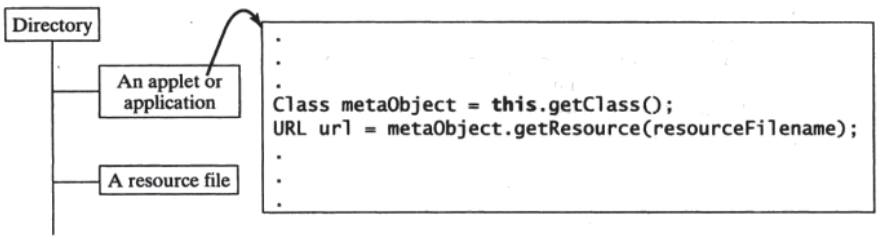
该方法会带来一个问题。文件的位置是固定的，因为它使用的是Windows平台的绝对文件路径。结果就造成它不能在其他平台上运行而且也不能作为applet运行。假设image/us.gif在类目录下。可以使用如下所示的相对路径来规避这个问题：

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

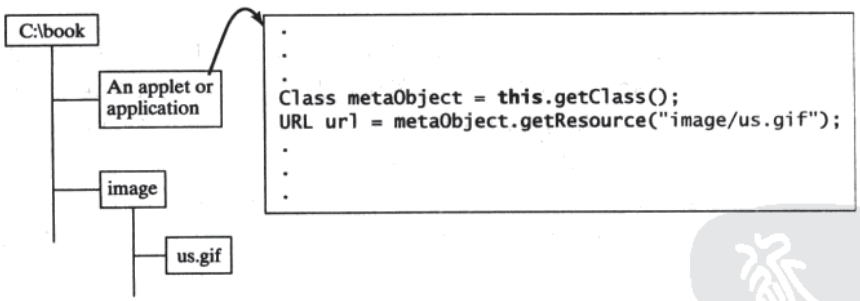
这样，Java应用程序就能在所有平台上正常工作，但是，这种方法对Java applet不起作用，因为applet不能加载本地文件。为了让它在应用程序和applet中都能工作，需要定位这个文件的URL。

java. net.URL类能够用来确定互联网上的文件（图像、音频和文本等）。笼统地说，URL（Uniform Resource Locator，统一资源定位符）是一个指向本地机器或远程主机上某个“资源”的指示器。这里的资源可以只是一个文件或一个目录。

如果文件放在类目录下，那么该文件的URL也可以以独立于文件位置的方式从一个类来访问。回顾一下，类目录就是存放类的位置。为了获取资源文件的URL，在applet或应用程序中使用下面的语句：



getClass()方法返回java.lang.Class类的一个实例。当每个类被载入内存时，Java虚拟机就会为该类创建一个实例。这个实例也称为元对象（meta object），它包括了关于类文件的信息，例如，类名、构造方法和方法。可以调用元对象上的getResource(filename)方法以获取类目录中一个文件的URL。例如，如果类文件在c:\book目录下，那么下面的语句就可以获取c:\book\image\us.gif的一个URL。



现在，可以使用下面的语句创建一个ImageIcon：

```
ImageIcon imageIcon = new ImageIcon(url);
```

程序清单18-11给出显示类目录中来自image/us.gif的图像的代码。文件image/us.gif在类目录下，而它的URL是使用getResource方法获取的（第5行）。一个有图像图标的标签在第6行创建。这个图像图标是从URL获取的。

程序清单18-11 DisplayImageWithURL.java

```
1 import javax.swing.*;
2
3 public class DisplayImageWithURL extends JApplet {
4   public DisplayImageWithURL() {
5     java.net.URL url = this.getClass().getResource("image/us.gif");
```

```
6      add(new JLabel(new ImageIcon(url)));
7    }
8  }
```

如果使用下面的代码替换第5～6行的代码：

```
add(new JLabel(new ImageIcon("image/us.gif")));
```

仍然可以独立地运行这个程序，但是不能在浏览器上运行它。

18.11 在任意Java程序中播放音频

音频文件有多种格式。Java程序能够播放WAV、AIFF、MIDI、AU和RMF格式的声音文件。

要在Java（应用程序或applet）中播放音频文件，应该先为声音文件创建一个音频剪辑对象（audio clip object）。一旦创建了音频剪辑，不需要重新加载文件就能重复播放声音。为了创建一个音频剪辑，使用java.applet.Applet类中的静态方法newAudioClip()：

```
AudioClip audioClip = Applet.newAudioClip(url);
```

声音原本是只能在Java applet中播放的。因为这个原因，AudioClip接口位于java.applet包中。从JDK 1.2开始，音频就能在任何一个程序中播放。

例如，下面的语句为类目录下的声音文件beep.au创建一个音频剪辑AudioClip对象：

```
Class metaObject = this.getClass();
URL url = metaObject.getResource("beep.au");
AudioClip audioClip = Applet.newAudioClip(url);
```

使用java.applet.AudioClip中的play()、loop()和stop()方法可以操控音频剪辑中的声音，如图18-14所示。

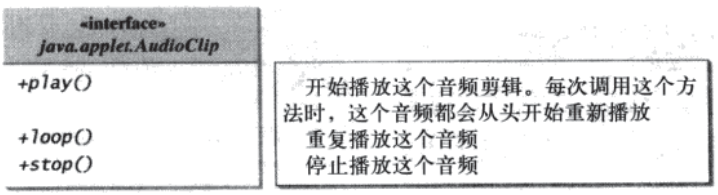


图18-14 AudioClip接口提供播放声音的方法

程序清单18-12给出显示丹麦国旗并重复播放丹麦国歌的代码。图像文件image/denmark.gif和音频文件audio/denmark.mid都存放在类目录下。第12行获取音频文件的URL。第13行为该文件创建一个音频剪辑对象。第14行重复播放这个音频文件。

程序清单18-12 DisplayImagePlayAudio.java

```
 1 import javax.swing.*;
 2 import java.net.URL;
 3 import java.applet.*;
 4
 5 public class DisplayImagePlayAudio extends JApplet {
 6   private AudioClip audioClip;
 7
 8   public DisplayImagePlayAudio() {
 9     URL urlForImage = getClass().getResource("image/denmark.gif");
10     add(new JLabel(new ImageIcon(urlForImage)));
11
12     URL urlForAudio = getClass().getResource("audio/denmark.mid");
13     audioClip = Applet.newAudioClip(urlForAudio);
14     audioClip.loop();
15   }
16
17   public void start() {
18     if (audioClip != null) audioClip.loop();
```

```
19   }
20
21   public void stop() {
22     if (audioClip != null) audioClip.stop();
23   }
24 }
```

当不显示applet时，stop方法（第21～23行）就会终止这个音频，当重新显示applet时，start方法（第17～19行）会重新启动这个音频。请尝试从浏览器运行该applet，并且观察一下没有stop和start方法的效果。

从主方法并在Web浏览器独立运行这个程序来测试它。回顾以前所讲的，为了简洁，所有applet中的main方法都没有在程序清单中打印出来。

18.12 实例学习：多媒体动画

本实例学习给出一个带图像和音频的多媒体动画。七个名为flag0.gif，flag1.gif，...，flag6.gif的图像分别是丹麦、德国、中国、印度、挪威、英国和美国七个国家的国旗。它们都存储在类路径的image目录下。音频包括了这七个国家的国歌anthem0.mid，anthem1.mid，...，anthem6.mid。它们都存储在类路径的audio目录下。

程序从第一个国家开始表示这些国家。对每个国家，程序显示它的国旗并播放它的国歌。当一个国家的音频结束时，就会出现下一个国家，依此类推。当显示完最后一个国家之后，程序会重新显示所有的国家。可以通过点击Suspend按钮暂停动画，点击Resume按钮重新开始播放动画，如图18-15所示。也可以直接从组合框中选择一个国家。

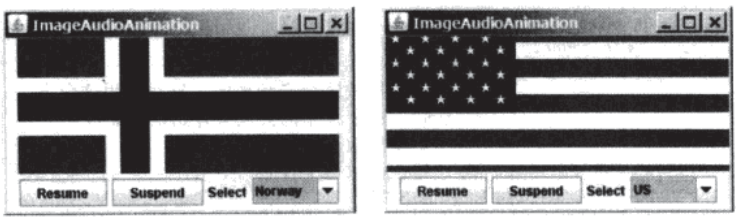


图18-15 applet显示一系列的图像并且播放音频

程序清单18-13给出这个程序。创建一个定时器来控制动画（第15行）。每次表示国旗的定时器延迟就是它的国歌的播放时间。可以使用RealPlayer或者Windows Media找到音频文件的播放时间。延迟时间存储在一个名为delays的数组中（第13～14行）。第一个音频文件（丹麦国歌）的延迟时间是48秒。

程序清单18-13 ImageAudioAnimation.java

```
 1 import java.awt.*;
 2 import java.awt.event.*;
 3 import javax.swing.*;
 4 import java.applet.*;
 5
 6 public class ImageAudioAnimation extends JApplet {
 7   private final static int NUMBER_OF_NATIONS = 7;
 8   private int current = 0;
 9   private ImageIcon[] icons = new ImageIcon[NUMBER_OF_NATIONS];
10   private AudioClip[] audioClips = new AudioClip[NUMBER_OF_NATIONS];
11   private AudioClip currentAudioClip;
12
13   private int[] delays =
14     {48000, 54000, 59000, 54000, 59000, 31000, 68000};
15   private Timer timer = new Timer(delays[0], new TimerListener());
16
17   private JLabel jlblImageLabel = new JLabel();
18   private JButton jbtResume = new JButton("Resume");
```

```
19  private JButton jbtSuspend = new JButton("Suspend");
20  private JComboBox jcboNations = new JComboBox(new Object[]
21    {"Denmark", "Germany", "China", "India", "Norway", "UK", "US"});
22
23  public ImageAudioAnimation() {
24    // Load image icons and audio clips
25    for (int i = 0; i < NUMBER_OF_NATIONS; i++) {
26      icons[i] = new ImageIcon(getClass().getResource(
27        "image/flag" + i + ".gif"));
28      audioClips[i] = Applet.newAudioClip(
29        getClass().getResource("audio/anthem" + i + ".mid"));
30    }
31
32    JPanel panel = new JPanel();
33    panel.add(jbtResume);
34    panel.add(jbtSuspend);
35    panel.add(new JLabel("Select"));
36    panel.add(jcboNations);
37    add(jlblImageLabel, BorderLayout.CENTER);
38    add(panel, BorderLayout.SOUTH);
39
40    jbtResume.addActionListener(new ActionListener() {
41      public void actionPerformed(ActionEvent e) {
42        start();
43      }
44    });
45    jbtSuspend.addActionListener(new ActionListener() {
46      public void actionPerformed(ActionEvent e) {
47        stop();
48      }
49    });
50    jcboNations.addActionListener(new ActionListener() {
51      public void actionPerformed(ActionEvent e) {
52        stop();
53        current = jcboNations.getSelectedIndex();
54        presentNation(current);
55        timer.start();
56      }
57    });
58
59    timer.start();
60    jlblImageLabel.setIcon(icons[0]);
61    jlblImageLabel.setHorizontalAlignment(JLabel.CENTER);
62    currentAudioClip = audioClips[0];
63    currentAudioClip.play();
64  }
65
66  private class TimerListener implements ActionListener {
67    public void actionPerformed(ActionEvent e) {
68      current = (current + 1) % NUMBER_OF_NATIONS;
69      presentNation(current);
70    }
71  }
72
73  private void presentNation(int index) {
74    jlblImageLabel.setIcon(icons[index]);
75    jcboNations.setSelectedIndex(index);
76    currentAudioClip = audioClips[index];
77    currentAudioClip.play();
78    timer.setDelay(delays[index]);
79  }
80
81  public void start() {
82    timer.start();
83    currentAudioClip.play();
```

```
84    }
85
86    public void stop() {
87      timer.stop();
88      currentAudioClip.stop();
89    }
90  }
```

第17行创建一个标签来显示国旗图像。第26～27行创建包含七个国旗图像的数组。第28～29行创建一个音频剪辑的数组。通过当前类的URL为每个音频文件创建一个音频剪辑。这些音频文件都存储在和applet类文件相同的目录中。

在第20～21行创建选择国家名的组合框。当在组合框中选择一个新国家名时，当前国家的表示停止，显示新选中的国家（第52～55行）。

`presentNation(index)`方法（第73～79行）表示一个带指定下标的国家。它在标签里设置一个新图像（第74行），和设置了选中下标的组合框来进行同步（第75行），播放新音频，并且设置新的延迟时间（第78行）。

applet的`start`和`stop`方法被覆盖以重新开始和暂停动画（第81～89行）。

关键术语

applet（Java小程序）
applet container（applet容器）
archive（存档）
HTML（超文本标记语言）
tag（标记）
signed applet（信任的applet）

本章小结

- `JApplet`是`Applet`的子类。它用于开发带Swing组件的Java applet。
- applet的字节码必须指定，使用HTML文件中的`<applet>`标记来告诉Web浏览器在何处能找到这个applet。applet可以使用`<param>`标记接收HTML文件中的字符串参数。
- applet容器通过`Applet`类中的`init`、`start`、`stop`和`destroy`方法控制和执行applet。
- 当加载applet时，applet容器调用一个applet的无参构造方法，创建一个applet的实例。创建applet后调用`init`方法。调用`init`方法后调用`start`方法。当再一次访问包含applet的页面再次激活applet时，也会调用`start`方法。当用户离开applet页面时，调用`stop`方法。
- 当浏览器正常退出时，就会调用`destroy`方法以通知applet不再需要它，并且应该释放它所占据的所有资源。`stop`方法总是在`destroy`方法之前调用。
- 编写应用程序和编写applet的过程非常相似。applet很容易转换成应用程序，反之亦然。此外，编写一个带主方法的applet可以使之独立运行。
- 在HTML中，可以使用applet标记中的`param`属性给applet传递参数。调用`getParameter(paramName)`方法可以获取参数值。
- 只有在创建了applet的一个实例之后，才可以调用`Applet`的`getParameter`方法。因此，这个方法不能被applet类的构造方法调用。只能从`init`方法中调用这个方法。
- 我们学习了如何将一个音频和一个图像放到applet和应用程序中。为了在applet和应用程序中加载音频和图像，必须创建音频和图像的URL。可以为类目录下的文件创建一个`URL`。
- 要播放音频，要用音频资源的URL创建一个音频剪辑。可以使用`AudioClip`类的`play()`方法播放一次，使用`loop()`方法重复播放，使用`stop()`方法停止播放。

复习题

18.2～18.6节

18.1 每个applet都是java.applet.Applet的一个实例吗？每个applet都是javax.swing.JApplet的一个实例吗？

18.2 描述Applet类中的init()、start()、stop()和destroy()方法。

18.3 怎样把组件添加到JApplet中？JApplet内容窗格的默认布局管理器是什么？

18.4 为什么图a中的applet什么都不显示？为什么图b中的applet在突出显示的行会抛出一个运行异常NullPointerException？

```
import javax.swing.*;

public class WelcomeApplet extends JApplet {
  public void WelcomeApplet() {
    JLabel jlblMessage =
      new JLabel("It is Java");
  }
}
```

a)

```
import javax.swing.*;

public class WelcomeApplet extends JApplet {
  private JLabel jlblMessage;

  public WelcomeApplet() {
    JLabel jlblMessage =
      new JLabel("It is Java");
  }

  public void init() {
    add(jlblMessage);
  }
}
```

b)

18.5 描述HTML的<applet>标记。如何向applet传递参数？

18.6 getParameter方法是在哪里定义的？

18.7 按如下修改DisplayMessage的applet，会发生什么错误？

```
public class DisplayMessage extends JApplet {
  /** Initialize the applet */
  public DisplayMessage() {
    // Get parameter values from the HTML file
    String message = getParameter("MESSAGE");
    int x =
      Integer.parseInt(getParameter("X"));
    int y =
      Integer.parseInt(getParameter("Y"));

    // Create a message panel
    MessagePanel messagePanel =
      new MessagePanel(message);
    messagePanel.setXCoordinate(x);
    messagePanel.setYCoordinate(y);

    // Add the message panel to the applet
    add(messagePanel);
  }
}
```

a）修改1

```
public class DisplayMessage extends JApplet {
  private String message;
  private int x;
  private int y;

  /** Initialize the applet */
  public void init() {
    // Get parameter values from the HTML file
    message = getParameter("MESSAGE");
    x = Integer.parseInt(getParameter("X"));
    y = Integer.parseInt(getParameter("Y"));
  }

  public DisplayMessage() {
    // Create a message panel
    MessagePanel messagePanel =
      new  MessagePanel(message);
    messagePanel.setXCoordinate(x);
    messagePanel.setYCoordinate(y);

    // Add the message panel to the applet
    add(messagePanel);
  }
}
```

b）修改2

18.8 应用程序和applet之间有何不同？如何运行一个应用程序？如何运行一个applet？应用程序和applet的编译过程是否相同？列出关于applet的一些安全限制。

18.9 能将一个框架放在一个applet中吗？

18.10 能将一个applet放在一个框架中吗？

18.11 在程序清单18-5的TicTacToe.java中，删除第97行的super.paintComponent(g)之后再运行程序，会发生什么现象？

18.9～18.10节

18.12 如何为类目录下的文件image/us.gif创建一个URL对象？

18.13 如何为类目录下的文件image/us.gif创建一个ImageIcon？

18.11节

18.14 Java中使用哪些类型的音频文件？

18.15 如何用类目录下的文件anthem/us.mid创建一个音频剪辑？

18.16 如何播放、重复播放以及停止播放音频剪辑？

编程练习题

教学注意 对练习题中的每个applet，都可以添加一个主方法以使它能够独立运行。

18.2～18.6节

18.1 （将应用程序转换成applet）将程序清单17-2转换成一个applet。

*18.2 （向applet传递字符串）改写程序清单18-5，显示带标准颜色、字体和字号的一条消息。message、x、y、color、fontname和fontsize都是<applet>标记的参数，如下所示：

```
<applet
  code = "Exercise18_2.class"
  width = 200
  height = 50
  alt = "You must have a Java-enabled browser to view the applet"
>
  <param name = MESSAGE value = "Welcome to Java" />
  <param name = X value = 40 />
  <param name = Y value = 50 />
  <param name = COLOR value = "red" />
  <param name = FONTNAME value = "Monospaced" />
  <param name = FONTSIZE value = 20 />
</applet>
```

18.3 （贷款计算器）编写一个applet计算贷款偿还额，如图18-16所示。用户可以输入利率、年数和贷款总额，然后点击Compute Payment按钮来显示月偿还额和总偿还额。

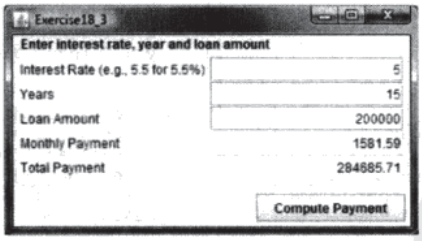


图18-16 applet计算贷款偿还额

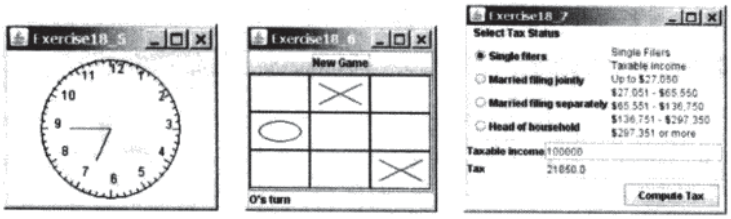
*18.4 （将应用程序转换成applet）将程序清单16-12中的ClockAnimation改写为一个applet并使之能够独立运行。

**18.5 （游戏：时钟学习工具）开发一个时钟applet，演示一年级学生如何读时钟。修改练习题15.19，在applet中显示一个带小时和分钟指针的详细时钟，如图18-17a所示。小时和分钟的值是随机产生的。小时的取值范围在0到11之间，分钟的取值是0、15、30或45。在时钟上点击鼠标就会显示新的随机时间。

**18.6 （游戏：井字游戏）做以下的修改来改写18.9节：

（1）将Cell声明为一个独立的类，而不是内部类。

（2）添加一个名为New Game的按钮，如图18-17b所示。这个New Game按钮会启动一个新游戏。

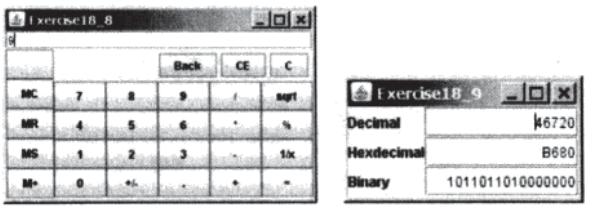


a) b) c)

图18-17 a) 在时钟上点击鼠标，就会随机显示一个时钟时间；b) 按钮New Game启动一个新游戏；c) 税款计算器根据指定的可征税收入和纳税人状况来计算税款

**18.7 (财务应用：税款计算器) 创建一个计算税款的applet，如图18-17c所示。这个applet允许用户选择纳税人状况，并且输入可征税收入，然后根据2001年美国联邦税率计算税款，如练习题10.8所示。

***18.8 (创建一个计算器) 使用FlowLayout、GridLayout和BorderLayout的面板设置下面的计算器，并实现加法 (+)、减法 (-)、除法 (/)、开平方 (sqrt) 和求余 (%) 的功能 (参见图18-18a)。



a) b)

图18-18 a) 练习题18.8是用Java实现一个流行的计算器；b) 练习题18.9实现十进制数、十六进制数和二进制数之间的转换

*18.9 (数的进制转换) 编写一个applet，实现十进制数、十六进制数和二进制数之间的转换，如图18-18b所示。在十进制数的文本域中输入一个十进制数之后点击回车键之后，就会在另外两个文本域中显示它对应的十六进制数和二进制数。类似地，也可以在其他文本域中输入值，然后进行相应的转换。

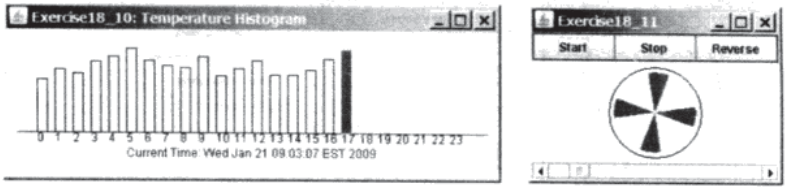
**18.10 (重绘部分区域) 重画面板的整个视图区域时，有时视图区域中只有一小部分区域发生了改变。只重画面板中受影响的部分是可以提高效率的。因为super.paintComponent(g)方法会导致视图区域全部清空，所以重绘面板时不能调用这个方法。采用部分重画的方式编写一个applet，在直方图中显示最近24小时中每个小时的温度。假设温度是在50～90华氏温度之间的随机温度，而且每小时更新一次。当前小时的温度需要重新显示，而其他各小时的温度图像保持不变。使用唯一的颜色突出显示当前小时的温度 (参见图18-19a)。

**18.11 (模拟：一个转动的风扇) 编写一个Java applet模拟转动的风扇，如图18-19b所示。按钮Start、Stop和 Reverse控制风扇。滚动条控制风扇的速度。创建JPanel的一个子类Fan来显示这个风扇。该类还包含暂停和重新启动风扇、设置风扇速度和颠倒转动方向的方法。创建一个名为FanControl的类，它包含一个风扇、三个按钮和一个控制风扇的滚动条。创建一个包含FanControl实例的Java applet。

**18.12 (控制一组风扇) 编写一个Java applet，在一组中显示三个风扇，用控制按钮来启动和停止整组风扇，如图18-20所示。

***18.13 (创建一个电梯模拟器) 编写一个applet，模拟电梯的上升和下降 (如图18-21所示)。左边的按钮表示乘客现在所在的楼层。乘客必须先点击左边的按钮，要求电梯到达他 (她) 所处的楼层。

进入电梯后，乘客点击右边的按钮指定想要到达的楼层。



a) b)

图18-19 a）直方图显示最近24小时内每小时的平均温度；b）程序模拟一个转动的风扇

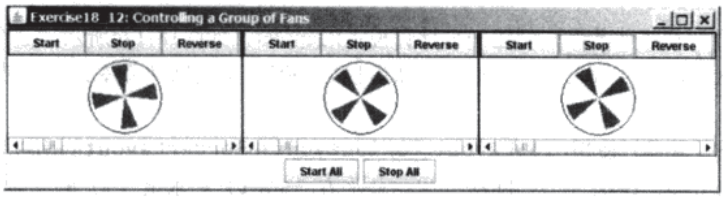


图18-20 程序运行和控制一组风扇

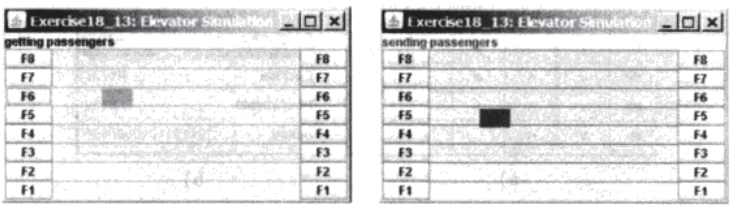


图18-21 程序模拟电梯的运转

*18.14 （控制一组时钟）编写一个Java applet，显示一组中的三个时钟，通过控制按钮来开始或停止它们，如图18-22所示。

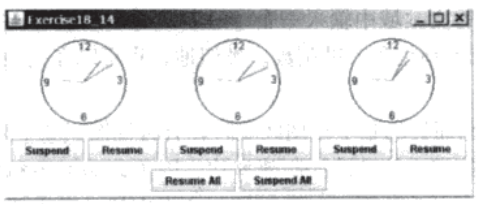


图18-22 三个独立运行的时钟，既可以单独控制又可以统一控制

18.10～18.12节

*18.15 （放大或缩小一个图像）编写一个applet，用同一个图像文件显示一系列大小不同的图像。初始状态时，这个图像的视图区域宽为300、高为300。程序应该使视图区域持续缩小到宽为50、高为50，每次宽和高都缩小1。达到这一状态后，视图区域应该持续扩大到宽为300、高为300，每次宽和高都增加1。这样，这个视图区域应该缩小和放大（交替地）一个图像以产生一个动画。

***18.16 （模拟股票行情）编写一个Java applet，显示股票指数行情（参见图18-23）。股票指数信息是从HTML文件中的<param>标记传递而来的。每个指数都有4个参数：指数名（例如，S&P 500）、当前时间（例如，15：54）、前一天的指数（例如，919.01）和变化量（例如，4.54）。至少使用5种指数，例如，道琼斯指数、标准普尔500指数、纳斯达克指数、日经指数和黄金与白银指数。用绿色表示上涨，红色表示下跌。这些指数在applet的视图区域中从右向左移动。当按下鼠标按钮时，这个applet就停在这个股票处；当放开按钮后，它又开始移动。

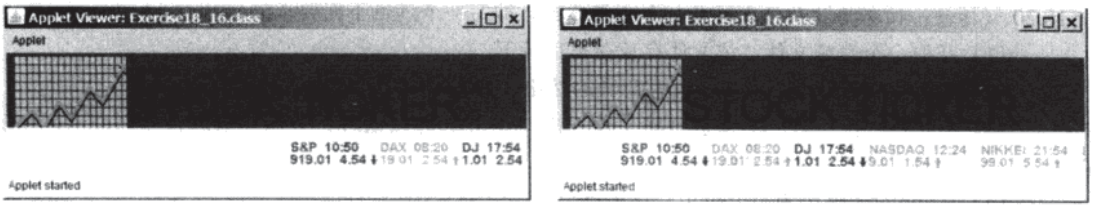
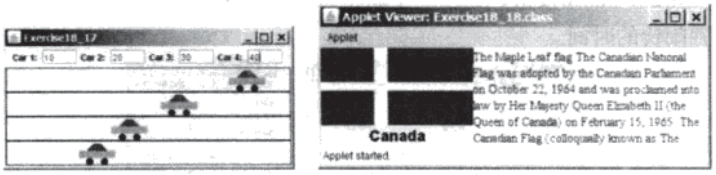


图18-23 程序显示股票指数行情

**18.17 （赛车）编写applet模拟四辆赛车，如图18-24a所示。可以对每辆赛车设置速度，用1表示最高速。



a) b)

图18-24 a）可以为每辆车设置速度；b）这个applet一个接一个地显示每个国家的国旗、名字以及简介，并且朗读当前显示的国家的简介

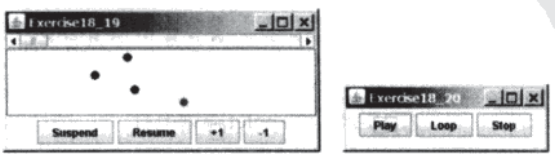
**18.18 （显示国旗）编写一个applet，通过展示每个国旗的图像、国名和简介一个接一个地介绍国旗（参见图18-24b），并且播放朗读简介的音频。

假设applet显示8个国家的国旗。这些照片图像文件的文件名为flag0.gif，flag1.gif，...，flag7.gif，都存储在applet目录下一个名为image的子目录中。每个音频的长度不超过10秒。假设每个国家的国名和国旗简介都是从HTML的参数传递得到的，参数分别为name0，name1，...，name7和description0，description1，...，description7。使用numberOfCountries将国家的个数作为HTML的参数来传递。下面是一个例子：

```
<param name = "numberOfCountries" value = 8>
<param name = "name0" value = "Canada">
<param name = "description0" value = "The Maple Leaf flag
The Canadian National Flag was adopted by the Canadian
Parliament on October 22, 1964 and was proclaimed into law
by Her Majesty Queen Elizabeth II (the Queen of Canada) on
February 15, 1965. The Canadian Flag (colloquially known
as The Maple Leaf Flag) is a red flag of the proportions
two by length and one by width, containing in its center a
white square, with a single red stylized eleven-point
maple leaf centered in the white square.">
```

提示 使用DescriptionPanel类显示图像、国名和文本。DescriptionPanel类在程序清单17-6中介绍。

***18.19 （弹跳的球）18.8节的例子模拟一个弹跳的球。扩展这个例子允许程序产生多个球，如图18-25a所示。可以使用+1或者-1按钮来增加或减少球的个数，并且使用Suspend和Resume按钮来暂停或重新开始球的跳动。对每个球随机地指定一种颜色。



a) b)

图18-25 a）applet允许添加或删除一个弹跳的球；b）点击Play播放音频剪辑一次，点击Loop会重复播放音频，而点击Stop会终止播放

*18.20 （播放、循环播放和停止播放一个音频剪辑）编写一个满足下面要求的applet：

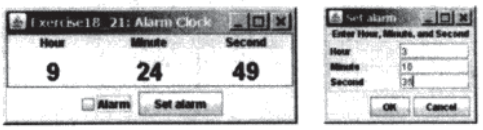
（1）获取一个音频文件，该文件存放在类目录下。

（2）放置三个标记为Play、Loop和Stop的按钮，如图18-25b所示。

（3）点击Play按钮时，会播放音频文件一次。点击Loop按钮时，会循环播放音频。点击Stop按钮时，停止播放该音频。

（4）该applet能够作为一个应用程序运行。

**18.21 （创建一个闹钟）编写一个applet，用一个较大的显示面板来显示小时、分钟和秒的数字钟。这个时钟允许用户设置闹铃时间。图18-26a给出这种时钟的一个例子。选中Alarm复选框可以启用闹铃。点击Set alarm按钮，显示一个新框架来指定闹铃的时间，如图18-26b所示。可以在该框架中设置闹铃的时间。



a) b)

图18-26 程序显示当前的小时、分钟和秒，并能设置闹铃

**18.22 （创建一个有声的图像动画）使用applet创建一个动画（参见图18-27），满足下面的要求：

- 允许用户指定动画速度。用户可以在文本域中输入速度。
- 用户输入帧数和图像文件名的前缀。例如，如果用户输入的帧数为n，图像文件名的前缀为L，那么图像文件就是L1，L2一直到Ln。假设这些图像都存储在image目录下，该目录是applet所在目录的子目录。
- 允许用户指定音频文件名。音频文件与applet存储在同一个目录下。动画开始时播放这个声音。

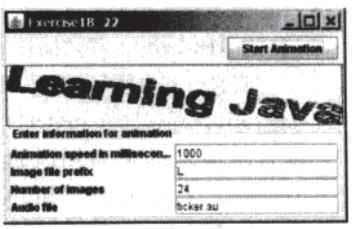


图18-27 这个applet允许用户选择图像文件、音频文件和动画速度

**18.23 （模拟：升旗并播放国歌）创建一个显示升国旗的applet，如图15-1所示。随着国旗的升起，播放国歌（可以使用程序清单18-13中的国旗图像和国歌音频文件）。

综合题

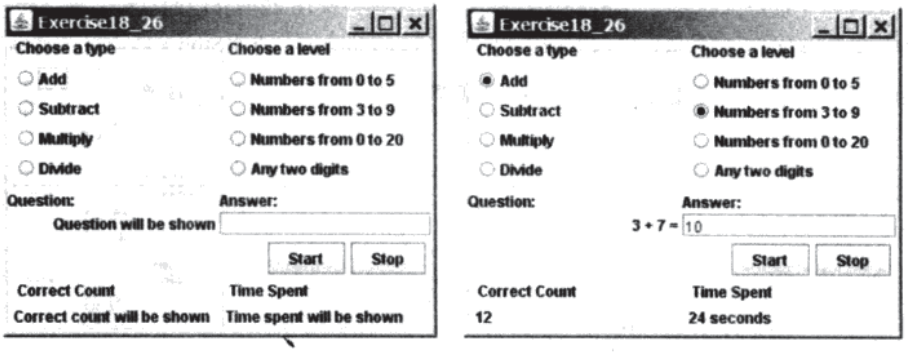
**18.24 （游戏：猜生日）程序清单3-3给出猜生日的程序。创建一个猜生日的applet，如图18-28所示。该applet提示用户检查生日是否在五个集合中的任意一个中。在点击Guess Birthday按钮之后文本域显示生日日期。

***18.25 （游戏：九宫格）7.7节介绍了九宫格问题。编写一个程序，让用户在applet的文本域中键入输入，如图18-1所示。点击Solve按钮显示结果。

图18-28 该applet猜测生日

***18.26 （游戏：数学测验）程序清单3-1和程序清单3-4，创建数学测验并给其打分。编写一个applet，允许用户选择题目的类型和难度，如图18-29a所示。当用户点击Start按钮时，程序开始生成题目。在用户输入答案和回车键之后，就会显示新的问题。当用户点击Start按钮时，就会显示所

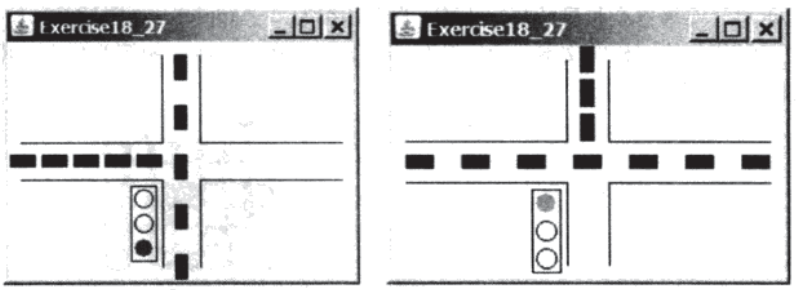
用的时间。每秒钟更新时间一次，直到点击Stop按钮。每当给出一个正确答案就更新正确答案的个数。



a）在任务开始之前　　b）在任务开始之后

图18-29　这个applet测试数学技巧

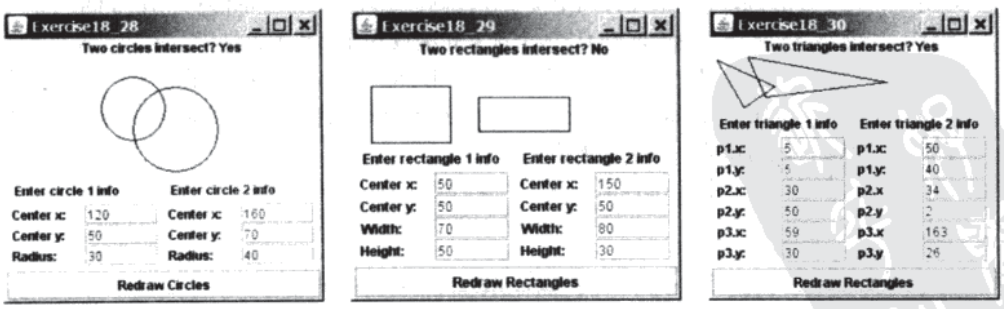
***18.27 （模拟：交通控制）练习题17.3使用单选按钮改变红绿灯。修改程序模拟十字路口的交通控制，如图18-30所示。当灯变为红色时，交通流是垂直的；当灯变为绿色的，交通流是水平的。灯每分钟都会自动变化一次。在灯从绿变为红之前，它首先会变为黄色，持续大约短短的五秒钟。



a）垂直的交通流　　b）水平的交通流

图18-30　这个applet模拟交通控制

**18.28 （几何方面：两个圆是否相交？）Circle2D类定义在练习题10.11中。编写一个applet，让用户确定圆的位置和大小，并且显示这两个圆是否相交，如图18-31a所示。



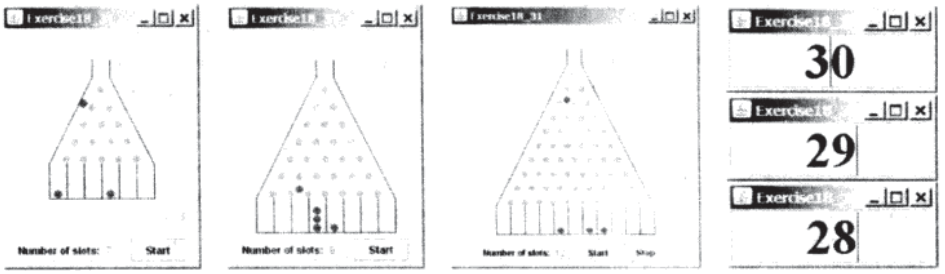
a）　　b）　　c）

图18-31　检查两个圆、两个矩形和两个三角形是否相交

**18.29 （几何方面：两个矩形是否相交？）MyRectangle2D类定义在练习题10.12中。编写一个applet，让用户确定矩形的位置和大小，并且显示这两个矩形是否相交，如图18-31b所示。

**18.30 （几何方面：两个三角形是否相交？）Triangle2D类定义在练习题10.13中。编写一个applet，让用户确定两个三角形的位置和大小，并且显示这两个三角形是否相交，如图18-31c所示。

***18.31 （游戏：豆机动画）编写一个applet，用动画实现练习题16.22所介绍的豆机。这个applet允许设置槽数，如图18-32所示。点击Start来启动或再次启动动画，点击Stop来停止这个动画。



a) b) c) d)

图18-32 a）～c）applet控制豆机的动画；d）applet进行时间的倒计时

*18.32 （倒计数警告）编写一个applet，允许用户在文本域中输入分钟数，然后点击回车键对分钟数进行倒计时，如图18-32d所示。每一分钟都会重新显示剩余的分钟数。当倒计时结束时，程序开始持续播放音乐。

**18.33 （模式识别：连续四个相同的数）为练习题7.19编写一个applet，如图18-33a和图18-33b所示。让用户在6行7列的网格的文本域中输入数字。如果存在一串四个相等的数字，用户就可以点击Solve按钮突出显示它们。

a) b) c)

图18-33 a）～b）点击Solve按钮突出显示在一行、一列或对角线上四个连续的数字；c）该applet是让两个玩家玩连接四子的游戏

***18.34 （游戏：连四子）练习题7.20允许两个玩家在控制台上玩连接四子的游戏。使用applet改写这个程序，如图18-33c所示。这个applet让两个玩家轮流放置红色和黄色棋子。为了放置棋子，玩家需要在可用的格子上点击。可用的格子（available cell）是指不被占用的格子，而其下方临接的格子是被占用的格子。如果一个玩家赢了，这个applet就闪烁这四个赢的格子，如果所有格子都被占用但还没有赢家，就报告无赢家。