# 20 Applets and HTML

# 20 Applets and HTML

*Welcome to the information superhighway.*

COMMON PHRASE

## Introduction

application
applet

Java programs are divided into two main categories, *applets* and *applications.* An **application** is an "ordinary" Java program while an **applet** is a kind of Java program that can be run across the Internet. Any Java program that is not an applet is an application program.

Applets are meant to be run from an HTML document. So, in Section 20.1 we give a very brief introduction to HTML. With many IDEs (Integrated Development Environments), it is possible to run an applet without bothering with HTML, via a program known as an *applet viewer.* If you are only interested in running applets via an applet viewer, you can skip Section 20.1 on HTML. In fact, you can skip this entire chapter with no loss of continuity in your reading. Nothing in the rest of this book depends on this chapter.

## Prerequisites

Chapters 17 and 18 cover regular Swing GUIs. This chapter covers applets, which is a popular but specialized topic within the general topic of Swing GUIs.

This chapter assumes that you have used a Web browser to read something on the Web, but does not assume that you know how to create documents to be viewed on the Web via a Web browser. Chapter 17 is also required for this chapter.

Sections 20.1 and 20.2 are independent of each other. You do not need Section 20.1 before reading Section 20.2. Section 20.3 requires both Sections 20.1 and 20.2.

## 20.1 A Brief Introduction to HTML

*You shall see them on a beautiful quarto page, where a neat rivulet of text shall meander through a meadow of margin.*

RICHARD BRINSLEY SHERIDAN, *The School for Scandal*

HTML

**HTML** is a language used to write documents that will be viewed on a Web browser, either over the Internet or entirely on your own computer. HTML is an acronym for

hypertext

**Hypertext Markup Language**. **Hypertext** is text that, when viewed on a browser,

links

contains clickable entries, called **links** or **hyperlinks**, which when clicked with your mouse display a different specified document. The documents themselves are often called **pages**, which is why a person's or a company's main location on the Internet is

home page

called a **home page**. The terms **HTML document** and **HTML page** mean the same thing and simply refer to a document created with the HTML language.

HTML is a very simple language. It consists of a collection of simple commands that you can insert into a text file to convert the file to a document meant to be viewed with a Web browser. The commands allow you to insert pictures and hyperlinks. They also allow you to write editing commands that specify what is a main heading, a subheading, a paragraph beginning, and so forth. Much of HTML is simply a language for formatting text, but HTML is not a word processor. It is more like a very simple programming language. HTML is very much like the annotations used by copy editors to mark a manuscript before it is typeset for production.

HTML is not part of the Java language, although, as you will see in Section 20.3, there can be some interaction between HTML and Java; namely, you can use HTML to display a Java applet program. We will give you only a small sample of the HTML language. This will allow you to design some simple documents for the Web (or just for your browser). If you want to become a more sophisticated user of HTML, you should consult a book dedicated entirely to HTML.

### HTML Formatting Commands

HTML consists of formatting commands that are added to ordinary text so that a browser knows how to display the text. There are two basic kinds of HTML commands: those that mark the beginning and end of a section of text, and those that mark a single location in the text.

Commands that mark the beginning and end of a section of text have the form

```
<Command>
Some text
</Command>
```

For example, the following makes the phrase "World's Greatest Home Page" a level 1 heading, which is the largest standard heading:

<h1>

```
<h1>
World's Greatest Home Page
</h1>
```

Notice that the notation `</`*`Command`*`>`, in this example `</h1>`, is used to mark the end of the text to which the command applies.

<h2>

You can have smaller heads, called level 2 heads (command `h2`), and even smaller heads, called level 3 heads (command `h3`), and so forth.

Commands that mark a single location in the text do not need to be closed with a command of the form `</`*`Command`*`>`. A good example of these kinds of commands is

<hr>

```
<hr>
```

which inserts a horizontal line, a good way to separate sections.

Commands in HTML are not absolute commands that determine the exact size of a portion of text or the exact location of line breaks. When you give a command for a level 1 head, you can reasonably assume that it will be larger than a level 2 head, but the browser will determine the exact size of the heading.

**line breaks**

The browser determines the line breaks in the displayed text, but you can force a line break by inserting the **break command**:

<br>

```
<br>
```

The browser will insert line breaks into displayed text where necessary to fit the text on the screen and to make the text "look nice." It will ignore your line breaks unless they are indicated with the `<br>` command.

<center>

You can make some layout specifications. For example, anything between the commands `<center>` and `</center>` will be centered on the page when it is displayed. The following will center the level 1 head we discussed earlier:

```
<h1>
<center>
World's Greatest Home Page
</center>
</h1>
```

Or, if you prefer, this code can also be written as follows:

```
<center>
<h1>
World's Greatest Home Page
</h1>
</center>
```

However, matching pairs, like `<h1>` and `</h1>` or `<center>` and `</center>`, should not cross. For example, the following is illegal (although some browsers may accept it):

```
<h1>
<center>
World's Greatest Home Page
</h1>
</center>
```

*This is illegal.*

**uppercase lowercase**

Unlike Java, HTML is not case sensitive. That is, HTML does not distinguish between uppercase and lowercase letters in commands. So, for example, `<p>` and `<P>` are the same command. We will write our HTML commands using lowercase letters, but they could just as well be written using uppercase letters. (Any text to be displayed as text to be read by the person viewing the document will, of course, display uppercase letters in uppercase and lowercase letters in lowercase.)

**file names**

An HTML file is a regular text file that you create and edit with a text editor in the same way that you write a Java program. HTML files should end with `.html`, but otherwise they can be named using the same rules you use to name other files on your system.

## OutLine of an HTML Document

**`<html>`** **`<head>`** **`<title>`**

Display 20.1 gives an outline for a simple HTML document. The entire document should be enclosed in the pair `<html>` and `</html>` at the beginning and end of the document. The head of the document is enclosed in `<head>` and `</head>`. The head is not displayed when the document is viewed; it records information that is used by a browser. In our outline, the only thing in the head is a title, enclosed in `<title>` and `</title>`. The title is used as a name for the document. For example, a browser will let you set a bookmark ("favorite") at a document so you can return to that document at a later time. The default name for the bookmark is the name given in this title.

**`<body>`**

The part of the document that is displayed on the screen is divided into two parts. The **body**, enclosed in `<body>` and `</body>`, is the real content of the document. The other displayed part is enclosed in `<address>` and `</address>`, and it is optional. It is supposed to contain the e-mail address for contacting the document's maintainer and the date that the document was last modified.

Display 20.2 shows an HTML document designed by following the outline in Display 20.1. Display 20.3 shows what this document will look like when viewed in a browser. Remember that the exact line breaks, size of letters, and other layout details are determined by the particular browser used, so it might look a bit different on your browser.

Display 20.1 **Outline of a Simple HTML Document**

```
<html> ←――――― Beginning of HTML document

<head> ←――――― Beginning of document head
<title> ←―――― Beginning of document title
Document Title
</title> ←――― End of document title
</head> ←―――― End of document head

<body> ←――――― Beginning of main text to appear on screen
<h1>
First of Largest Size Headings
</h1>
Some text                Most browsers do not require all the entries pointed to
<h2>                     with blue arrows, but it is good to include them all.
First Subheading
</h2>
Some text
<h2>
Second Subheading
</h2>
Some text.
<h1>
Second of Largest Size Headings
</h1>
More of the same
   .
   .
   .
</body> ←―――― End of main text to appear on screen

<address> ←―― Beginning of address section
<hr> ←――――――― Horizontal line; nice, but not required
The e-mail address of the person maintaining the page.
Also, the date of the last time the page was changed.
(You can put in whatever you want here, but the ←――― Although this is not part of the
 e-mail address and date are what people expect.)     main body of text, it does appear
</address> ←― End of address section                  on the screen.

</html> ←――――― End of HTML document
```

Display 20.2 **An HTML Document**

```
<html>
<head>
<title>
Liars Club Home Page
</title>
</head>

<body>
<h1>
<center>
Liars Club
</center>
</h1>

<h2>
Club Goals
</h2>
<p>
The goal of the club is to take over the world.
We already have members in key government positions.
</p>
<p>
Another goal is to improve the image of liars.
To this end, we have infiltrated many advertising agencies.
</p>

<h2>
Meeting Times
</h2>
The first Saturday of each month at 5 AM.
<p> <!--To add some space.-->
</p>

</body>

<address>
<hr>
webmaster@epimenides.org
<br>
June 1, 1888
</address>
</html>
```

*This text does not appear on the screen but is used as the default name for any bookmark ("favorite") to this page.*

*Blank lines are ignored when the document is displayed, but they can make your HTML code easier to read.*

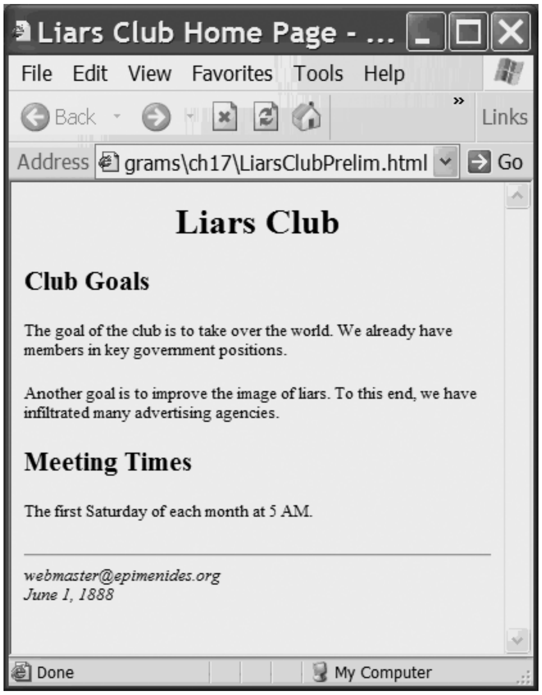
*Text may have different line breaks when displayed on your browser.*

*A new paragraph will always produce a line break and some space.*

*A comment*

*This is the file* **`LiarsClubPrelim.html`**.

Display 20.3 **Browser View of Display 20.2**


Liars Club Home Page - ...
File Edit View Favorites Tools Help
Back
Links
Address grams\ch17\LiarsClubPrelim.html
Go
Done
My Computer

Liars Club

Club Goals

The goal of the club is to take over the world. We already have members in key government positions.

Another goal is to improve the image of liars. To this end, we have infiltrated many advertising agencies.

Meeting Times

The first Saturday of each month at 5 AM.

*webmaster@epimenides.org*
*June 1, 1888*

## TIP: Comments

comment

The following illustrates how you write comments in HTML:

```
<!--This is a comment-->
```

Any text between the four symbols <!-- and the three symbols --> is considered a comment. A sample of a comment is given in Display 20.2. ■

## Hyperlinks

hyperlink

Anybody who has spent even a little time surfing the Web knows that you can click things on a Web page that send you to a different page. These things you click are called **links** or **hyperlinks**. In this subsection we will show you how to mark text as a hyperlink so that if the user clicks that text the browser goes to a specified other Web page.

The syntax for hyperlinks is as follows:

```
<a href="Path_To_Document">
Text_To_Click
</a>
```

For example, the following creates a link to the Sun's Java home page:

```
<a href="http://java.sun.com">
Sun Microsystems Java website
</a>
```

Be sure to include the quotation marks as shown. The `Text_To_Click` will be displayed and underlined (or otherwise highlighted) by the browser. In this example, if the person viewing the document clicks the text `Sun Microsystems Java website`, then the browser will display Sun's Java home page.

In Display 20.4 we have enhanced the HTML document from Display 20.2 by including a link to the Web site of a different kind of "liars" organization. If you view this HTML document with a browser, it will look approximately like Display 20.5. If you click the text that is underlined, the browser will display the Web page located at

```
http://liars.org/
```

Other enhancements to Display 20.4 will be discussed in subsequent subsections.

### URL

A **URL** is the name of an HTML document on the Web. URL is an acronym for **Uniform Resource Locator**. For example:

```
http://liars.org/
```

URLs often begin with `http`, which is the name of the protocol used to transfer and interpret the HTML document. Most browsers will allow you to omit the beginning part `http://` and will fill it in for you.

## Inserting a Picture

The command to insert a picture in an HTML document is as follows:

```
<img src="File_With_Picture">
```

For example, suppose you have a digital version of a picture in the subdirectory (subfolder) `pictures`. To be concrete, suppose the picture file `sunset.gif` is in the directory (subfolder) `pictures`, and `pictures` is a subdirectory (subfolder) of the directory

Display 20.4 An HTML Document with a Hyperlink and a Picture

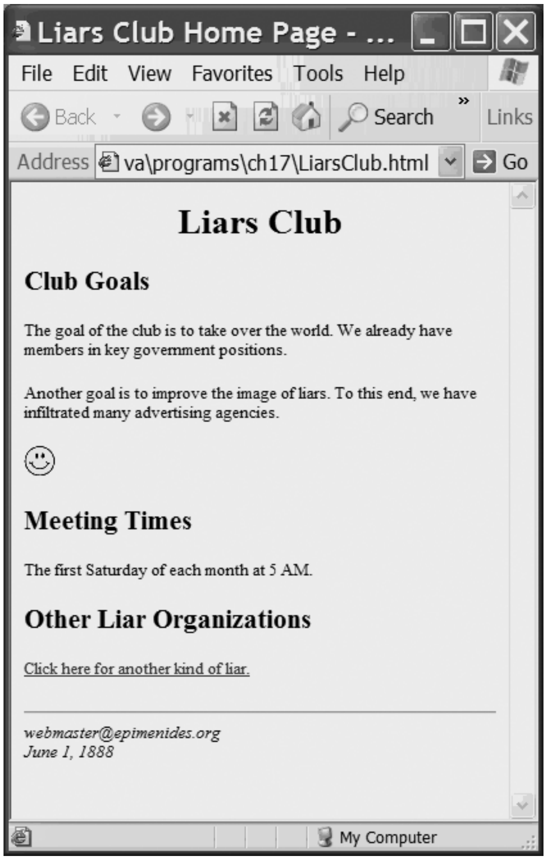
```
<html>
<head>
<title>
Liars Club Home Page
</title>
</head>

<body>
<h1>
<center>
Liars Club
</center>
</h1>
<h2>
Club Goals
</h2>
<p>
The goal of the club is to take over the world.
We already have members in key government positions.
</p>
<p>
Another goal is to improve the image of liars.
To this end, we have infiltrated many advertising agencies.
</p>
<img src="smiley.gif">
<h2>
Meeting Times
</h2>
The first Saturday of each month at 5 AM.
<h2>
Other Liar Organizations
</h2>
<a href="http://liars.org/">
Click here for another kind of liar.
</a>
<p> <!--To add some space.-->
</p>
</body>

<address>
<hr>
webmaster@epimenides.org
<br>
June 1, 1888
</address>
</html>
```

*This is the file* `LiarsClub.html`*.*

*This is explained in the subsection entitled "Inserting a Picture."*

*A hyperlink to another HTML document, in this case another Web page.*

Display 20.5 **Browser View of Display 20.4**


Liars Club Home Page - ...
File Edit View Favorites Tools Help
Back Search Links
Address va\programs\ch17\LiarsClub.html Go
My Computer

Liars Club

Club Goals

The goal of the club is to take over the world. We already have members in key government positions.

Another goal is to improve the image of liars. To this end, we have infiltrated many advertising agencies.

Meeting Times

The first Saturday of each month at 5 AM.

Other Liar Organizations

Click here for another kind of liar.

*webmaster@epimenides.org*
*June 1, 1888*

(folder) where the HTML page is located. You could add the picture to your HTML document by inserting the following:

```
<img src="pictures/sunset.gif">
```

You can use either a full path name or a relative path name to the file with the encoded picture (that is, relative to the directory containing the HTML document). So, if the picture file is in the same directory (same folder) as the HTML document, then you simply use the picture file name. Most commonly used picture-encoding formats are accepted. In particular, `.gif`, `.tiff`, and `.jpg` files are accepted.

Displays 20.4 and 20.5 show an example of an HTML document with a picture of a smiley face inserted.

### Inserting a Hyperlink

The command for inserting a hyperlink in an HTML document is as follows:

**SYNTAX**

```
<a href="Path_To_Document">
Text_To_Click
</a>
```

**EXAMPLE**

```
<a href="http://java.sun.com">
Sun Microsystems Java Website
</a>
```

The *`Path_To_Document`* can be either a full or relative path name to an HTML file or a URL to any place on the Web. If the person viewing the document clicks the *`Text_To_Click`*, the document indicated by *`Path_To_Document`* will be displayed.

### Inserting a Picture in an HTML Document

The command for inserting a digital picture in an HTML document is as follows:

**SYNTAX**

```
<img src="File_With_Picture">
```

**EXAMPLE**

```
<img src="pictures/sunset.gif">
```

The *`File_With_Picture`* can be either a full or relative path name to a file with a digitally encoded picture. Most commonly used picture-encoding formats are accepted. In particular, `.gif`, `.tiff`, and `.jpg` files are accepted.

## PITFALL: Not Using Your Browser's Refresh Command

Browsers normally keep copies of the most recently used HTML pages. That way, if you want to go back to one of those pages, the browser can quickly retrieve the page. However, when you are designing and debugging an HTML page, this feature can be a problem.

Suppose you test an HTML page with your browser and notice something that needs to be fixed. If you change the HTML page to fix the problem and then look again at the page with your browser, you will undoubtedly see no change. This can be

**PITFALL:** (continued)

true even if you exit your browser, reenter the browser starting with some other page, and then jump to the page being fixed. This is because the browser keeps copies of the most recent pages that were displayed and uses those copies rather than any more recently changed copies.

Browsers have a command, which may be a button or a menu item, that causes the browser to reload a page and thus get the most recent version of the page. This button or menu item is likely to be called “Refresh” or “Reload.” To ensure that your browser is displaying the most recent version of a page, click this refresh command. ■

## Self-Test Exercises

1. When a browser displays text from an HTML document, where does the browser put the line breaks? Does it leave them where they are in the HTML document? Does it insert a line break after every 80 characters? Or does it do something else?
2. What is the difference between the commands `<br>` and `<BR>`?
3. How do you insert a link to the following home page in an HTML document? The browser should go to this Web page when the text `"Click for tango."` is clicked.

   ```
   http://www.elmundodeltango.com/
   ```
4. How do you insert a link to the file `goodstuff.html` in an HTML document? The browser should go to the HTML document `goodstuff.html` when the user clicks `"Click for good stuff."`. The file `goodstuff.html` is in the subdirectory (subfolder) named `stuff`, which is a subdirectory (subfolder) of the directory (folder) that contains the HTML document.

**TIP: Other Languages for Authoring Web Pages ★**

HTML is a low-level language. HTML for a Web browser is analogous to assembly language for a computer. Most Web page designers now use a high-level Web page design language that translates into HTML. Three examples of such high-level languages are Dreamweaver (Macromedia, Inc.), FrontPage (Microsoft Corporation), and GoLive (Adobe Systems Inc.). ■

## 20.2 Programming Applets

*I thought applets were those jellied candies made from apples.*

STUDENT COMMENT

On first reading, the word *applet* may suggest a small apple, but it is meant to suggest a small *application*. (Recall that a Java *application* is an ordinary Java program.) The term comes from the fact that applets were intended to be used for small programs run over the Internet. However, there are no size constraints on applets. In this section we will describe how you write applets and how you can view them without any connection to the Internet. In Section 20.3 we will show you how to run an applet over the Internet.

An applet is very much like a Swing GUI, and if you understand some details about the regular Swing GUIs we discussed in Chapter 17, then you will find it very easy to write applets. Applets and regular (non-applet) Swing GUIs are not disjoint topics. Any significant applet will use material from what we have already covered for regular Swing GUIs. If you go on to learn more about Swing, you can use almost all of your new Swing techniques in your applets. However, you need not know about applets to write regular (non-applet) Swing GUIs.

### Defining an Applet

An applet class is normally defined as a derived class of the class `JApplet`. Display 20.6 shows the position of the class `JApplet` in the class hierarchy. Note that the class `JApplet` is in the package `javax.swing`. The class `Applet`, which also appears in Display 20.6, is an older class for constructing applets that has been superseded by the `JApplet` class.

`JApplet`

You can design an applet class as a derived class of `JApplet` in much the same way as you defined regular Swing GUIs as derived classes of `JFrame`. Display 20.7 contains a very simple example of an applet class. As illustrated there, an applet uses the method `init` to do the initializations that we would do in a constructor for a regular Swing GUI. An applet class definition normally defines no constructors.

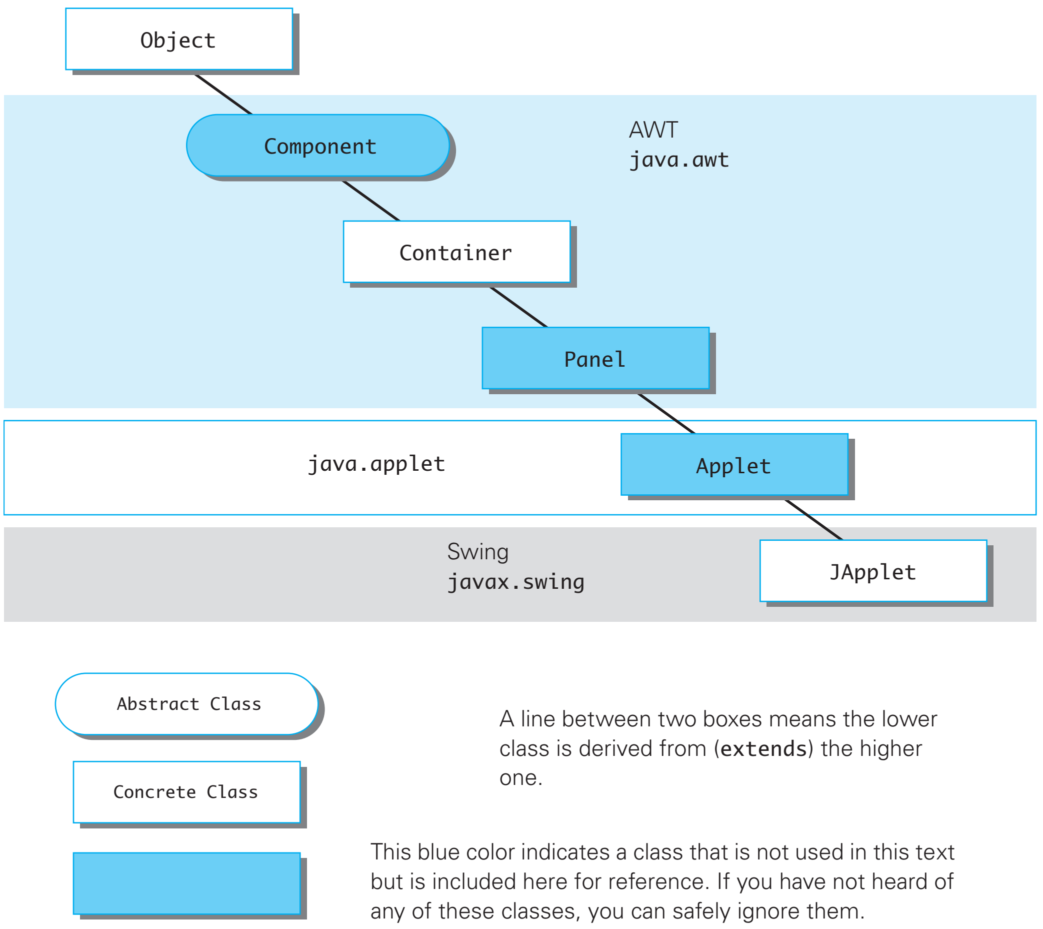
`init`

You use the method `add` to add components to an applet in the same way that you add components to a `JFrame`.

Some of the items you are used to including in a regular Swing GUI are not included in an applet definition. Applets do not use the `setVisible` method and applets normally do not contain a demonstration `main` method. Applets are displayed automatically by a Web page or an applet viewer.

Applets do not have titles, so the `setTitle` method is not used in an applet. Applets also do not use the `setSize` method. Applets are normally embedded in an HTML document, and the HTML document can add any desired title. As you will see in Section 20.3, the HTML document also takes care of sizing the applet.

Display 20.6 Applets in the Class Hierarchy


Object
Component
AWT
java.awt
Container
Panel
java.applet
Applet
Swing
javax.swing
JApplet
Abstract Class
Concrete Class
A line between two boxes means the lower class is derived from (extends) the higher one.
This blue color indicates a class that is not used in this text but is included here for reference. If you have not heard of any of these classes, you can safely ignore them.

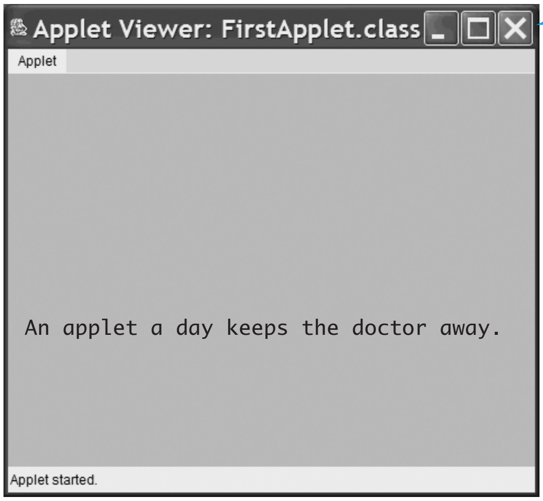
Applets do not have a close-window button and so do not have a `setDefault-CloseOperation` method. When the HTML document containing the applet is closed, that automatically closes the applet. When run using an applet viewer, it will look like the applet has a close-window button, but that is the close-window button of the applet viewer.

Display 20.7 An Applet

```
import javax.swing.JApplet;
import javax.swing.JLabel;
import java.awt.BorderLayout;
import java.awt.Color;

public class FirstApplet extends JApplet
{
    public void init()
    {
        getContentPane().setBackground(Color.ORANGE);
        setLayout(new BorderLayout());
        JLabel aLabel =
             new JLabel("An applet a day keeps the doctor away.");
        add(aLabel, BorderLayout.CENTER);
    }
```

*The `init()` method is used instead of a constructor.*

**Resulting GUI** (Using an Applet Viewer)



```
An applet a day keeps the doctor away.
```

*This close-window button and the other two buttons are part of the applet viewer, not part of the applet.*

### The `JApplet` Class

The `JApplet` class is the class normally used to create an applet. This class is in the `javax.swing` package.

## Running an Applet

You compile an applet class in the same way that you compiled all of the other Java classes you have seen. However, you run an applet differently from other Java programs. The normal way to run an applet is to embed it in an HTML document. The applet is then run and viewed through a Web browser. We will discuss this way of viewing an applet in Section 20.3. Applets can also be viewed using an **applet viewer**, a program designed to run applets as stand-alone programs. If you are using an IDE (Integrated Development Environment) that has a menu command called Run Applet, Run, Execute, or something similar, you can probably use one of these commands to run an applet just as you run an ordinary Java application program. (In the TextPad environment the command is Run Java Applet on the Tools menu. This environment command will automatically invoke an applet viewer. If a window pops up asking you to Choose a file, answer "No.") If this does not work, you will need to check with a local expert or else read Section 20.3 for details on running an applet by embedding it in an HTML document.

applet viewer

If you run the applet shown in Display 20.7 using an applet viewer, the result will look similar to the GUI shown in that display.

If you cannot run an applet viewer from an IDE, you can undoubtedly run an applet viewer with a one-line command to your operating system. For example, the applet in Display 20.7 would be run as follows:

```
appletviewer FirstApplet.html
```

However, if you run an applet with a one-line command in this way, you may (or may not) need to create an HTML document yourself (named `FirstApplet.html` in this example) and place the applet in the HTML document. HTML documents and how to place an applet in an HTML document are described in Section 20.3.

## Menus in a `JApplet`

Menus are constructed and added to a `JApplet` just as they are for a `JFrame`. In particular, a `JApplet` has a method named `setJMenuBar` that behaves the same as the `setJMenuBar` method of a `JFrame`. As with a `JFrame`, a `JApplet` can also have menu bars added to the `JApplet` or to a panel that is part of the `JApplet` using the `add` method. See Self-Test Exercise 8 for an example of a `JApplet` with a menu.

## TIP: Converting a Swing Application to an Applet

The fastest and easiest way to describe how you define an applet is to tell you how you need to modify an ordinary Swing GUI to transform it into an applet. The details are as follows:

1. Derive the class from the class `JApplet` instead of from the class `JFrame`. That is, replace `extends JFrame` with `extends JApplet`.
2. Remove the `main` method. An applet does not need the things that are typically placed in `main`.

(continued)

**TIP:** (continued)

3. Replace the constructor with a no-parameter method named `init`. The body of the `init` method can be the same as the body of the deleted constructor, but with some items removed. The items you need to remove are described in the following steps.
4. Delete any invocation of `super`. The `init` method is not a constructor so `super` cannot be used.
5. Delete any method invocations that serve to program the close-window button of a windowing GUI. An applet has no close-window button. (The close-window button you see when running an applet viewer is part of the applet viewer, not part of the applet.) This means you delete any invocation of `setDefaultCloseOperation`. In Chapter 17 we will discuss registering window listeners with the method `addWindow-Listener`. Window listeners respond to window events such as clicking the close-window button. So, you also delete any invocation of `addWindowListener`.
6. Delete any invocation of `setTitle`. Applets have no titles.
7. Delete any invocation of `setSize`. Sizing is done by the applet viewer or by the HTML document in which the applet is embedded.

For example, the applet in Display 20.8 was obtained from the Swing GUI in Display 19.19 by following these rules. ■

## Self-Test Exercises

5. Does a `JApplet` have a `setJMenuBar` method? If it does have a `setJMenuBar` method, does it serve the same function as the `setJMenuBar` method of a `JFrame`?
6. Where in the definition of an applet do you normally do initializations such as adding buttons to the applet? In a constructor? In some method? If so, which method?
7. Why is there normally no `main` method in an applet definition?
8. Convert the `JFrame` in Display 17.14 to a `JApplet`.

Display 20.8 **An Applet Calculator** (part 1 of 4)

```
import javax.swing.JApplet;
import javax.swing.JTextField;
import javax.swing.JPanel;
import javax.swing.JLabel;
import javax.swing.JButton;
import java.awt.BorderLayout;
import java.awt.FlowLayout;
import java.awt.Color;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
```

Display 20.8 **An Applet Calculator** (part 2 of 4)

```
/**
 A simplified calculator as an applet.
 The only operations are addition and subtraction.
*/
public class AppletCalculator extends JApplet
                              implements ActionListener
{
    public static final int WIDTH = 400;
    public static final int HEIGHT = 200;
    public static final int NUMBER_OF_DIGITS = 30;

    private JTextField ioField;
    private double result = 0.0;


    public void init()
    {
        setLayout(new BorderLayout());

        JPanel textPanel = new JPanel();
        textPanel.setLayout(new BorderLayout());
        ioField =
             new JTextField("Enter numbers here.", NUMBER_OF_DIGITS);
        ioField.setBackground(Color.WHITE);
        textPanel.add(ioField);
        add(textPanel, BorderLayout.NORTH);

        JPanel buttonPanel = new JPanel();
        buttonPanel.setBackground(Color.BLUE);
        buttonPanel.setLayout(new FlowLayout());

        JButton addButton = new JButton("+");
        addButton.addActionListener(this);
        buttonPanel.add(addButton);
        JButton subtractButton = new JButton("-");
        subtractButton.addActionListener(this);
        buttonPanel.add(subtractButton);
        JButton resetButton = new JButton("Reset");
        resetButton.addActionListener(this);
        buttonPanel.add(resetButton);

        add(buttonPanel, BorderLayout.CENTER);
    }
```

*We deleted the* **main** *method.*

*We deleted invocations of* **setSize**, **setTitle**, *and* **setDefaultCloseOperation**.

*Code on this page is identical to the corresponding code in Display 17.19.*

*The method* **actionPerformed** *is identical to the one in Display 17.19.*

(continued)

Display 20.8 **An Applet Calculator** (part 3 of 4)

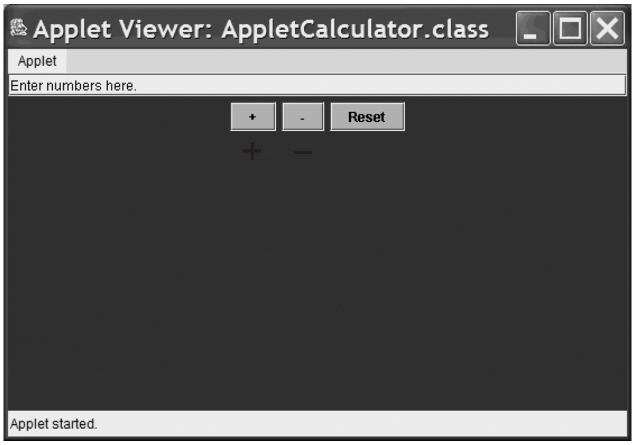
```
    public void actionPerformed(ActionEvent e)
    {
        try
        {
            assumingCorrectNumberFormats(e);
        }
        catch (NumberFormatException e2)
        {
            ioField.setText("Error: Reenter Number.");
        }
    }
```

*The methods* **`assumingCorrectNumberFormats`** *and* **`stringToDouble`** *are identical to the ones in Display 17.19.*

```
    //Throws NumberFormatException.
    public void assumingCorrectNumberFormats(ActionEvent e)
    {
        String actionCommand = e.getActionCommand();
        if (actionCommand.equals("+"))
        {
            result = result + stringToDouble(ioField.getText());
            ioField.setText(Double.toString(result));
        }
        else if (actionCommand.equals("-"))
        {
            result = result - stringToDouble(ioField.getText());
            ioField.setText(Double.toString(result));
        }
        else if (actionCommand.equals("Reset"))
        {
            result = 0.0;
            ioField.setText("0.0");
        }
        else
            ioField.setText("Unexpected error.");
     }


    //Throws NumberFormatException.
    private static double stringToDouble(String stringObject)
    {
        return Double.parseDouble(stringObject.trim());
    }

}
```

Display 20.8 **An Applet Calculator** (part 4 of 4)

**Resulting GUI** (When started)



## Adding Icons to an Applet ★

Display 20.9 illustrates the use of an icon in an applet. An icon is simply a picture. It is typically, but not always, a small picture. The easiest way to display an icon in an applet is to place the icon in a `JLabel`. In the applet in Display 20.9, the picture in the file `duke_waving.gif` is displayed as an icon that is part of the `JLabel` named `aLabel`. The three lines that create the label, create the icon, and add the icon to the label are repeated below:

```
JLabel aLabel = new JLabel("Welcome to my applet.");
ImageIcon dukeIcon = new ImageIcon("duke_waving.gif");
aLabel.setIcon(dukeIcon);
```

**ImageIcon**

The class `ImageIcon` is in the `javax.swing` package. The icon picture is a digital picture in one of the standard formats. The picture, in this case `duke_waving.gif`, must be converted to an `ImageIcon` before it can be added to a label. This is done as follows: `new ImageIcon("duke_waving.gif")`. So, the following creates an `ImageIcon` based on the picture `duke_waving.gif` and stores a reference to the icon in the variable `dukeIcon`:

```
ImageIcon dukeIcon = new ImageIcon("duke_waving.gif");
```

**setIcon**

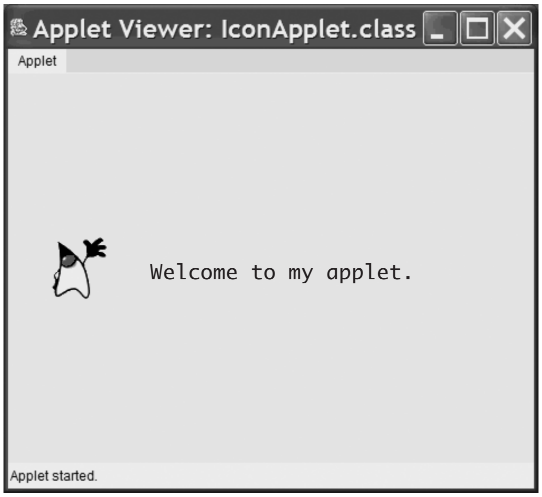
The method `setIcon` adds an icon to a label, as in the following:

```
aLabel.setIcon(dukeIcon);
```

Display 20.9 An Applet with an Icon

```
import javax.swing.JApplet;
import javax.swing.JLabel;
import javax.swing.ImageIcon;
import java.awt.BorderLayout;
import java.awt.Color;

public class IconApplet extends JApplet
{
    public void init()
    {
        getContentPane().setBackground(Color.YELLOW);
        setLayout(new BorderLayout());

        JLabel shift = new JLabel("             ");
        JLabel aLabel = new JLabel("Welcome to my applet.");
        ImageIcon dukeIcon = new ImageIcon("duke_waving.gif");
        aLabel.setIcon(dukeIcon);
        add(shift, BorderLayout.WEST);
        add(aLabel, BorderLayout.CENTER);
    }
}
```

**Resulting GUI**[1]



Welcome to my applet.

Applet started.

[1] Java, Duke, and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

The character pictured in this icon is named *Duke*. He is Sun Microsystem's mascot for the Java language.

If you want the label to have only the icon and no text, then simply use the no-argument constructor when you create the JLabel. For example, if in Display 20.9 we had used

```
JLabel aLabel = new JLabel();
```

instead of

```
JLabel aLabel = new JLabel("Welcome to my applet.");
```

then no string would appear in the applet; only the icon would be displayed.

Icons can be used in this way in JFrames and JPanels. Their use is not restricted to applets. Chapter 18 contains more material on icons.

### Icons

An **icon** is a picture, typically, but not always, a small picture. The icon is given as a file in any of a number of standard formats (such as gif, tiff, or jpg). The class ImageIcon is used to convert a picture file to a Swing icon so the icon can be used as a component to add to any Container class, such as a JApplet.

**SYNTAX**

```
ImageIcon Name_Of_ImageIcon =
        new ImageIcon("Picture_File_Name");
```

The *Picture_File_Name* is either a relative or absolute path name to the picture file. (So if the picture file is in the same directory [same folder] as your applet, you need give only the name of the picture file.)

**EXAMPLE**

```
ImageIcon dukeIcon = new ImageIcon("duke_waving.gif");
```

## Self-Test Exercises

9. How do you add an icon based on the digital picture file myPicture.gif to an applet?

10. Must the digital picture file used to create an ImageIcon be in the gif format?

11. When you specify the file for a digital picture, such as duke_waving.gif, as an argument to the ImageIcon constructor, can you use a path name?

## 20.3 Applets in HTML Documents

*Write it here; run it there!*

SOUNDS LIKE AD COPY[2]

In this section we describe how you place an applet in an HTML document so that when the document is viewed by a user at a Web browser anywhere in the world, the applet is displayed and run.

### Inserting an Applet in an HTML Document

**applet tag**

You can place an applet in an HTML document with a command known as an **applet tag**. For example, the following applet tag will display the applet in Display 20.8:

```
<applet code="AppletCalculator.class" width=400 height=300>
</applet>
```

This applet tag assumes that the HTML file and the file `AppletCalculator.class` are in the same directory (same folder). If they were not in the same directory, then you would use an absolute or relative path name for the class `AppletCalculator.class`.

Display 20.10 contains an HTML document that includes the applet tag we just gave. When displayed with a browser, this HTML document would look approximately as shown in Display 20.11.

**sizing an applet**

Notice that when you place an applet in an HTML document, you give the name of the `.class` file for the applet, not the `.java` file. Also notice that you specify the width and height of the applet in this command and not within the applet class definition. The width and height are given in pixels.

#### Applet Tag

The command for inserting an applet in an HTML document is called an **applet tag** and is written as described below:

**SYNTAX**

```
<applet code="Path_To_Applet" width=Number_1 height=Number_2>
</applet>
```

**EXAMPLE**

```
<applet code="AppletCalculator.class" width=400 height=300>
</applet>
```

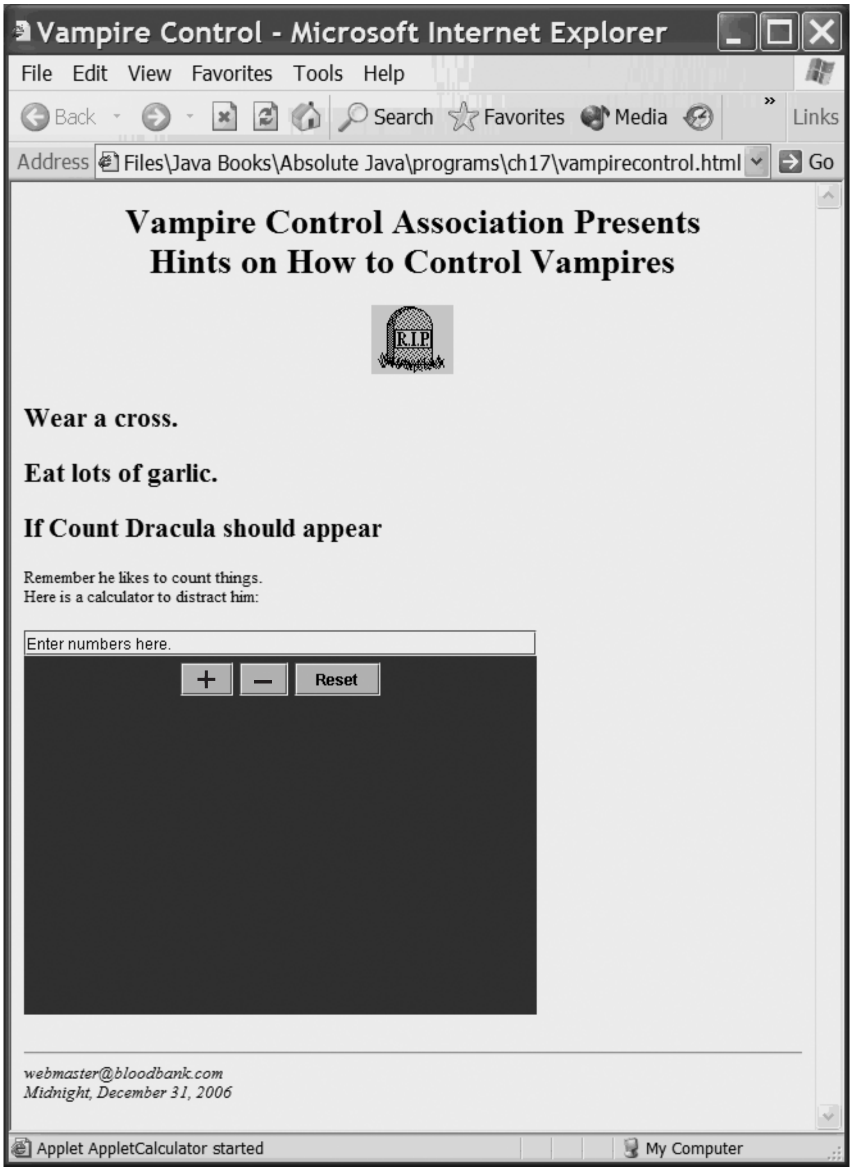
The *`Path_To_Applet`* can be either a full or relative path name to the applet's `.class` file. *`Number_1`* and *`Number_2`* are integers giving the width and height of the displayed applet in pixels.

[2] But I just made it up.

Display 20.10 An Applet in an HTML Document

```
<html>
<head>
<title>
Vampire Control
</title>
</head>

<body>
<center>
<h1>
Vampire Control Association Presents
<br>
Hints on How to Control Vampires
</h1>
<img src="RIP.gif">
</center>

<h2>
Wear a cross.
</h2>
<h2>
Eat lots of garlic.
</h2>
<h2>
If Count Dracula should appear
</h2>
Remember he likes to
count things.
<br>
Here is a calculator to distract him:
<p> <!--To add some space.-->
</p>

        <applet code="AppletCalculator.class" width=400 height=300>
        </applet>

<p> <!--To add some space.-->
</p>
</body>

<address>
<hr>
webmaster@bloodbank.com
<br>
Midnight, December 31, 2006
</address>
</html>
```

Display 20.11 **Browser View of Display 20.10**


Vampire Control - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media Links
Address Files\Java Books\Absolute Java\programs\ch17\vampirecontrol.html Go

**Vampire Control Association Presents**
**Hints on How to Control Vampires**

**Wear a cross.**

**Eat lots of garlic.**

**If Count Dracula should appear**

Remember he likes to count things.
Here is a calculator to distract him:

Enter numbers here.

*webmaster@bloodbank.com*
*Midnight, December 31, 2006*

Applet AppletCalculator started
My Computer

## Running an Applet over the Internet ★

When you run an applet in an HTML document and a user at a different location views that HTML document over the Internet, the `.class` file for the applet is sent over the Internet to enable that user to run the applet on his or her Web browser. This is possible because Java is so portable. A `.class` file produced on one computer will run on any other computer that has a Java Virtual Machine installed in its Web browser. (Recall that a Java Virtual Machine is the interpreter used when you run the byte-code in a Java `.class` file.) Note that it is the Java Virtual Machine in the browser that is used to run the applet. Essentially all Web browsers now come equipped with a Java Virtual Machine. So, as long as the browser is installed, the computer on which the browser is run need not have any other Java software such as a Java compiler.

### Self-Test Exercises

12. When you specify an applet in an applet tag for an HTML document, do you give the `.java` file or the `.class` file?
13. Give the HTML code to insert the applet named `CoolApplet` into an HTML document.
14. ★ Suppose you have Java installed on your computer. This installation includes a Java Virtual Machine that is used when you run Java application programs. Your Web browser also has its own Java Virtual Machine. So, your computer has two Java Virtual Machines. When you use your browser to run an applet embedded in an HTML page, which Java Virtual Machine is used?

## PITFALL: Using an Old Web Browser

Web browsers do not use the same Java Virtual Machine that is used to run regular Java applications. If you have an old Web browser, it will have an old Java Virtual Machine (or, if it is very old, no Java Virtual Machine). Earlier Java Virtual Machines do not know about `JApplets`, since an older class used to be used to create applets. Thus, if you have an old browser, you may not be able to run applets (of the kind discussed here) from an HTML document. This can be true even if Java applications run fine on your system. The solution for this problem is to obtain a new browser.

**applet viewer**

Even if you may have problems running your applets on a browser, you should have no such problem running them from the applet viewer, as long as you have a recent version of Java. So, you should be able to run and test your applets, even if you cannot run them from an HTML page. However, if your applets are not ultimately placed in HTML documents, you may as well use regular Swing GUIs derived from `JFrame`.

## Applets and Security ★

Suppose somebody on the Web reads your HTML page and that HTML page contains an applet. That applet's byte-code is run on the browser that is on the reader's computer. So, your applet can be a program that runs on other people's computers. Similarly, other people's applets can run on your computer.

Whenever somebody else's program runs on your computer, there are serious security concerns. Will the program leave a virus on your computer? Will it read confidential information from your files? Will it corrupt your operating system? Applets are designed so that they cannot (or at least cannot easily) do any of these things. Applets cannot run any of your programs, and they cannot read or write to files on your computer.

### Self-Test Exercises

15. ★ When somebody on machine A views an HTML document that is on machine B and that contains an applet, is the applet running on machine A or machine B?

16. ★ Can an applet open a file for writing to the file? Can an applet open a file for reading from the file?

## Chapter Summary

- Documents designed to be read by a Web browser can be written in a language called HTML. HTML is an acronym for Hypertext Markup Language.
- Applets are Java programs designed to be placed in and run from an HTML document.
- Applets are similar to Swing GUIs derived from the class `JFrame`.
- An applet is normally a derived class of the class `JApplet`.
- An applet normally has no `main` method and no constructors. However, the method `init` serves the same purpose as a constructor for an applet.

## Answers to Self-Test Exercises

1. The browser ignores line breaks in the document and inserts line breaks so that the text fits on the screen and "looks good." However, it will insert a line break at the location of a `<br>` command and at certain other commands such as `<p>` and `</p>`.

2. There is no difference. HTML is not case sensitive.

3.
   ```
<a href="http://www.elmundodeltango.com/">
Click for tango.
</a>
```

4.
   ```
<a href="stuff/goodstuff.html">
Click for good stuff.
</a>
```

5. A `JApplet` does have an instance variable of type `JMenuBar` and a method `setJMenuBar`. These are the same for a `JApplet` as they are for a `JFrame`.
6. In the `void` method `init()`.
7. The things you are likely to do in a main method, such as make the applet visible, are done in the Web page containing the applet (or in the applet viewer).
8. Just follow the rules we gave in the last subsection for converting a `JFrame` to a `JApplet`. The resulting class is shown below and is given in the file `AppletMenuDemo.java` on the accompanying CD.

**extra code on CD**

```
import javax.swing.JApplet;
import javax.swing.JPanel;
import java.awt.GridLayout;
import java.awt.Color;
import javax.swing.JMenu;
import javax.swing.JMenuItem;
import javax.swing.JMenuBar;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

public class AppletMenuDemo extends JApplet
                            implements ActionListener
{
    public static final int WIDTH = 300;
    public static final int HEIGHT = 200;

    private JPanel redPanel;
    private JPanel whitePanel;
    private JPanel bluePanel;

    public void init()
    {
        setLayout(new GridLayout(1, 3));

        redPanel = new JPanel();
        redPanel.setBackground(Color.LIGHT_GRAY);
        add(redPanel);

        whitePanel = new JPanel();
        whitePanel.setBackground(Color.LIGHT_GRAY);
        add(whitePanel);
```

```
            bluePanel = new JPanel();
            bluePanel.setBackground(Color.LIGHT_GRAY);
            add(bluePanel);

            JMenu colorMenu = new JMenu("Add Colors");

            JMenuItem redChoice = new JMenuItem("Red");
            redChoice.addActionListener(this);
            colorMenu.add(redChoice);

            JMenuItem whiteChoice = new JMenuItem("White");
            whiteChoice.addActionListener(this);
            colorMenu.add(whiteChoice);

            JMenuItem blueChoice = new JMenuItem("Blue");
            blueChoice.addActionListener(this);
            colorMenu.add(blueChoice);

            JMenuBar bar = new JMenuBar();
            bar.add(colorMenu);
            setJMenuBar(bar);
        }

        public void actionPerformed(ActionEvent e)
        {
            String buttonString = e.getActionCommand();

            if (buttonString.equals("Red"))
                 redPanel.setBackground(Color.RED);
            else if (buttonString.equals("White"))
                whitePanel.setBackground(Color.WHITE);
            else if (buttonString.equals("Blue"))
                bluePanel.setBackground(Color.BLUE);
            else
                System.out.println("Unexpected error.");
        }
    }
```

9. You create a `JLabel`. You create the icon from `myPicture.gif` using a constructor for the class `ImageIcon`. You add the icon to the label with the method `setIcon`. For example, the following might be used in the `init` method of an applet:

```
JLabel someLabel = new JLabel();
ImageIcon myIcon = new ImageIcon("myPicture.gif");
someLabel.setIcon(myIcon);
add(someLabel);
```

10. No. In particular, it can be in the `tiff` or `jpg` format among others.

11. You may use a relative or absolute path name when specifying the picture file.
12. The `.class` file.
13. 
```
<applet code="CoolApplet.class" width=400 height=300>
</applet>
```

    The values of `400` and `300` may, of course, be replaced with other values.
14. The Java Virtual Machine in your Web browser.
15. Machine A.
16. No, applets cannot read from or write to files.

## Programming Projects

myCodeMate *Many of these Programming Projects can be solved using AW's CodeMate. To access these please go to:* www.aw-bc.com/codemate.

1. Using any three different images of your choice, write an applet that displays the first image in a `JLabel`. Add a `JButton` that displays the next image when pressed. Cycle back to the first image when the last image is displayed. The code provided with this book contains three sample images named `image1.gif`, `image2.gif`, and `image3.gif` for you to use.
2. myCodeMate Redo or do for the first time the tic-tac-toe project (Programming Project 2 in Chapter 17, but do it as an applet.
3. myCodeMate Convert the Swing GUI in Display 17.17 to an applet and place it in an HTML document.
4. Redo or do for the first time Programming Project 6 in Chapter 17, but do it as an applet.
5. Redo or do for the first time Programming Project 7 in Chapter 17, but do it as an applet.
6. Redo or do for the first time Programming Project 8 in Chapter 17, but do it as an applet.