OO for GUI Design (contd.)

Java Applets

A `JApplet` in Swing and an `Applet` in AWT are graphical objects that can be embedded in a web page and viewed at a remote site when a web browser points to the web page.

Practically any graphical application in Java can be converted into an applet.

An applet is an AWT/Swing component, just like a button, a text area, etc., with the difference that an applet is meant to be executed by a special version of the Java Virtual Machine that is embedded in a Java-enabled web browser.

There are basically four steps to go through to convert a graphical application that is invoked from a command line into an applet that is invoked automatically by a web browser at a remote workstation:

1. Change the header of the class so that the class now extends `JApplet` or `Applet` as opposed to, say, `JFrame` or `Frame`. If you plan to use Swing components in your applet, you must extend `JApplet`.

2. The code that was in the constructor is placed in the `init()`. What a constructor does for a regular Java graphical program, `init()` does for an applet – it carries out the initializations needed when the applet is first loaded in.

3. Eliminate the function `main()`.

4. Declare the applet class public so that the program that runs the applet, which will either be browser JVM or the `appletviewer` tool, can access it.

Life Cycle of an Applet:

The `Applet` class provides the following methods that can be used to control the different phases of an applet:

`init()`

`start()`

`stop()`

`destroy()`

The Applet Tag:

To embed an applet in a web page, you need to invoke the applet class within the `<APPLET>` and `</APPLET>` HTML tags. As shown below, the `<APPLET>` tag comes with a number of attributes, some of which are mandatory and other optional:

```
< APPLET

    [CODEBASE = codebaseURL]

    CODE = appletFile

    [ALT = alternateText]

    [NAME = appletInstanceName]

    WIDTH = pixels

    HEIGHT = pixels

    [ALIGN = alignment]

    [VSPACE = pixels]

    [HSPACE = pixels]

>

[< PARAM NAME = appletParameter1 VALUE = value >]

[< PARAM NAME = appletParameter2 VALUE = value >]

. . .

[alternateHTML]

</APPLET>
```

```
String boxWidthString = getParameter( "BOX_WIDTH" );
if ( boxWidthString != null )
  boxWidth = Integer.parseInt( boxWidthString );
else boxWidth = 200;
```

An Applet Example:

1. The applet should display a sequence of images to a user who has invoked the applet at a remote site. The nature of the display should be as if the user was watching a slide show presentation. For that reason, the applet will be called `SlideShowApplet`.

2. Since it can take a while to load all the images into the browser Java Virtual Machine, the applet should display status images like *"Loading image i"* for different value of i as the ith image is loaded in.

3. The images should be displayed a fixed time interval apart. The time interval between successive images should be a parameter value supplied by the HTML code associated with the applet.

4. If an image is larger than the size of the applet window as specified in the HTML file, then the image should appear with scrollbars so that all parts of the image can be examined.

- Use the `Timer` class to display the images at fixed intervals.

- Load images into the program in a separate thread of computation.

- Use the `ImageIcon` class (as opposed to the `Image` class for representing the images.

- Use a `JLabel` as a `Scrollable` object for displaying scrollable images when the image size is larger than the size of the applet window.

```
Timer timer;
....
....
timer = new Timer( pause, new ActionListener() {
            public void actionPerformed( ActionEvent evt ) {
                ....
                ....
                contentPane.repaint();
            }
        });
timer.setInitialDelay( 0 );
timer.setCoalesce(false);
....
....
timer.start();
....
....
timer.stop();
....
....
timer.restart();
```

```
new Thread() {
     public void run() {
         loadImages();
     }
}.start();
```

```
ImageIcon[] images[ numImages ];
....
....
JScrollPane scrollableImage;
....
....
scrollableImage =
      new JScrollPane( new JLabel( images[ frameIndex - 1 ],
                          JLabel.CENTER ) );
scrollableImage.setPreferredSize( new Dimension( displayWidth,
                                    displayHeight - 8 ) );
```

```
public void paintComponent( Graphics g ) {
       super.paintComponent( g );
       if ( ..... ) {
           scrollableImage =
                     new JScrollPane(
                         new JLabel( images[ frameIndex - 1 ],
                                     JLabel.CENTER ) );
               scrollableImage.setPreferredSize(
                 new Dimension( displayWidth, displayHeight - 8 ) );
            }
            if ( scrollableImage != null ) {
                contentPane.removeAll();
                contentPane.add( scrollableImage );
            }
            contentPane.revalidate();
            contentPane.setVisible( true );
            ....
}
```

```
public void loadImages() {
    String prefix = dir + "/flower";
    for ( int i = 0; i < numImages; i++ ) {
        ....
        ....
        try {
          images[i] = new ImageIcon( new URL( getCodeBase() +
                                    prefix + (i+1) + ".jpg" ) );
        } catch( MalformedURLException m ) {
            System.out.println("Couldn't create image: badly formed URL" );
        }
    }
    ....
    ....
}
```

```
images[i] = new ImageIcon(
         new URL( getCodeBase() + prefix + (i+1) + ".jpg" ) );
```

```
 String imageFileName;
 ImageIcon image = new ImageIcon( imageFileName );
```

```
//SlideShowApplet.java

import javax.swing.*;
import java.awt.*;              //for Graphics, Color, Dimension, etc.
import java.awt.event.*;
import java.net.*;              //for URL needed for image loading

public class SlideShowApplet extends JApplet {

    int frameIndex = 0;         //the current frame number
    String dir;                 //the directory relative to the codebase
    Timer timer;                //the timer for sequencing through the images
    int pause;                  //the time interval between images
    int numImages;              //number of images to display
    int width;                  //width of the applet's content pane
    int height;                 //height of the applet's content pane
    int displayWidth;
    int displayHeight;
    JComponent contentPane;     //the applet's content pane
    ImageIcon images[];         //the images
    boolean finishedLoading = false;
    JLabel statusLabel;
    JScrollPane scrollableImage;
    boolean newFrameAvailable = false;


    public void init() {
        //Get the applet parameters.
        String at = getParameter("dir");
        dir = (at != null) ? at : "images/slideshow";
        at = getParameter("pause");
        pause = (at != null) ? Integer.valueOf(at).intValue() : 2000;
        at = getParameter("numImages");
        numImages = (at != null) ? Integer.valueOf(at).intValue() : 10;
        width = getWidth();
        height = getHeight();
```

```
displayWidth = width - getInsets().left - getInsets().right;
displayHeight = height - getInsets().top - getInsets().bottom;

contentPane = new JPanel() {
    public void paintComponent( Graphics g ) {
        super.paintComponent( g );
        if ( finishedLoading && newFrameAvailable ) {
            scrollableImage =
                 new JScrollPane(
                     new JLabel( images[ frameIndex - 1 ],
                                 JLabel.CENTER ) );
            scrollableImage.setPreferredSize(
              new Dimension( displayWidth, displayHeight - 8 ) );
        }
        if ( scrollableImage != null ) {
            contentPane.removeAll();
            contentPane.add( scrollableImage );
        }
        contentPane.revalidate();
        contentPane.setVisible( true );
        newFrameAvailable = false;
    }
};

contentPane.setBackground(Color.white);
setContentPane(contentPane);

statusLabel = new JLabel("Loading Images...", JLabel.CENTER);
statusLabel.setForeground( Color.red );
contentPane.add(statusLabel);
```

```
    timer = new Timer( pause, new ActionListener() {
            public void actionPerformed( ActionEvent evt ) {
                frameIndex++;
                if ( frameIndex == numImages )
                    frameIndex = 1;
                newFrameAvailable = true;
                contentPane.repaint();
            }
    });

    timer.setInitialDelay( 0 );
    timer.setCoalesce(false);

    images = new ImageIcon[numImages];

    new Thread() {
            public void run() {
                loadImages();
            }
    }.start();
}



public void start() {
    if ( finishedLoading )
        timer.restart();
}


public void loadImages() {
    String prefix = dir + "/flower";
    for ( int i = 0; i < numImages; i++ ) {
        statusLabel.setText( "loading image " + ( i + 1 ) );
        try {
          images[i] = new ImageIcon( new URL( getCodeBase() +
```

```
                                        prefix + (i+1) + ".jpg" ) );
            } catch( MalformedURLException m ) {
                System.out.println("Couldn't create image: badly formed URL" );
            }
        }
        finishedLoading = true;
        statusLabel.setText( null );
        timer.start();
    }


    public void stop() {
        timer.stop();
    }


    public String getAppletInfo() {
        return "Title: A SlideShow Applet\n";
    }

    public String[][] getParameterInfo() {
        String[][] info = {
          {"dir", "String", "the directory containing the images to loop"},
          {"pause", "int", "the time interval between successive frames"},
          {"numImages", "int", "the number of images to display; default is 10
        };
        return info;
    }
}
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<html>
<head>
<title>Run Slide Show</title>

<body>
<APPLET  CODE = "SlideShowApplet.class"  WIDTH = "600" HEIGHT = "500" >
<PARAM NAME = "numImages" VALUE ="21">
<PARAM NAME = "pause" VALUE ="8000">
<PARAM NAME = "dir" VALUE ="images/slideshow">
Your browser is completely ignoring the &lt;APPLET&gt; tag!
</APPLET>

</body>
</html>
```

```
appletviewer SlideShowApplet.html
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<html>
<head>
<title>Run Slide Show</title>
<body>

<!--"CONVERTED_APPLET"-->
<!-- CONVERTER VERSION 1.3 -->
<OBJECT classid="clsid:8AD9C840-044E-11D1-B3E9-00805F499D93"
WIDTH = "600" HEIGHT = "500" codebase=
  "http://java.sun.com/products/plugin/1.3/jinstall-13-win32.cab#Version=1,3,0,
<PARAM NAME = CODE VALUE = "SlideShowApplet.class" >

<PARAM NAME="type" VALUE="application/x-java-applet;version=1.3">
<PARAM NAME="scriptable" VALUE="false">
<PARAM NAME = "numImages" VALUE  ="21">
<PARAM NAME = "pause" VALUE  ="8000">
<PARAM NAME = "dir" VALUE  ="images/slideshow">
<COMMENT>
<EMBED type="application/x-java-applet;version=1.3"  CODE =
  "SlideShowApplet.cl\ass" WIDTH = "600" HEIGHT = "500" numImages = "21"
  pause = "8000" dir = "images/slideshow"  scriptable=false
  pluginspage="http://java.sun.com/products/plugin/1.3/plugin-install.html"><NO
Your browser is completely ignoring the &lt;APPLET&gt; tag!
</NOEMBED></EMBED>
</OBJECT>

<!--
<APPLET CODE = "SlideShowApplet.class" WIDTH = "600" HEIGHT = "500">
<PARAM NAME = "numImages" VALUE  ="21">
<PARAM NAME = "pause" VALUE  ="8000">
<PARAM NAME = "dir" VALUE  ="images/slideshow">
Your browser is completely ignoring the &lt;APPLET&gt; tag!

</APPLET>
-->
```

```
<!--"END_CONVERTED_APPLET"-->

</body>
</html>
```

Dual-Purpose Programming for Applets:

```
public static void main( String[] args )    // WRONG
{
  SlideShowApplet s = new SlideShowApplet();
  s.setSize( 600, 500 );
  s.setVisible( true );
}
```

```
public static void main( String[] args ) {
    JFrame f = new JFrame( "slide show" );
    ....
    ....
    Container cPane = f.getContentPane();
    SlideShowApplet2 slideshow = new SlideShowApplet2();
    cPane.add( slideshow, BorderLayout.CENTER );
    f.setSize( 600, 500 );
    f.setVisible( true );
}
```

```
//SlideShowApplet2.java

import javax.swing.*;
import java.awt.*;              //for Graphics, Color, Dimension, etc.
import java.awt.event.*;
import java.net.*;              //for URL

public class SlideShowApplet2 extends JApplet {

    boolean inApplet = true;    //needed to switch between execution as
                                //as applet and as a stand-alone application

    int frameIndex = -1;        //the current frame number
    String dir;                 //the directory relative to the codebase
    Timer timer;                //the timer for sequencing through the images
    int pause;                  //the time interval between images
    int numImages;              //number of images to display
    int width;                  //width of the applet's content pane
    int height;                 //height of the applet's content pane
    int displayWidth;
    int displayHeight;
    JComponent contentPane;     //the applet's content pane
    ImageIcon images[];         //the images
    boolean finishedLoading = false;
    JLabel statusLabel;
    JScrollPane scrollableImage;
    boolean newFrameAvailable = false;

    //constructor needed to run the applet as a stand-alone application
    public SlideShowApplet2() {
        inApplet = false;
        init();
    }

    public void init() {
        //check if running as an applet or as a stand-alone application
```

```
if ( inApplet ) {
    String at = getParameter("dir");
    dir = (at != null) ? at : "images/slideshow";
    at = getParameter("pause");
    pause = (at != null) ? Integer.valueOf(at).intValue() : 15000;
    at = getParameter("numImages");
    numImages = (at != null) ? Integer.valueOf(at).intValue() : 21;
    width = getWidth();
    height = getHeight();
}
else {
    dir = "images/slideshow";
    pause = 15000;
    numImages = 21;
    width = 600;
    height = 500;
}

displayWidth = width - getInsets().left - getInsets().right;
displayHeight = height - getInsets().top - getInsets().bottom;

contentPane = new JPanel() {
    public void paintComponent( Graphics g ) {
        super.paintComponent( g );
        if ( finishedLoading && newFrameAvailable ) {
            scrollableImage =
                 new JScrollPane(
                     new JLabel( images[ frameIndex ],
                                 JLabel.CENTER ) );
            scrollableImage.setPreferredSize(
              new Dimension( displayWidth, displayHeight - 8 ) );
        }
        if ( scrollableImage != null ) {
            contentPane.removeAll();
            contentPane.add( scrollableImage );
        }
```

```
            contentPane.revalidate();
            contentPane.setVisible( true );
            newFrameAvailable = false;
        }
    };

    contentPane.setBackground(Color.white);
    setContentPane(contentPane);

    statusLabel = new JLabel("Loading Images...", JLabel.CENTER);
    statusLabel.setForeground( Color.red );
    contentPane.add(statusLabel);

    timer = new Timer( pause, new ActionListener() {
            public void actionPerformed( ActionEvent evt ) {
                frameIndex++;
                if ( frameIndex == numImages )
                    frameIndex = 0;
                newFrameAvailable = true;
                contentPane.repaint();
            }
    });
    timer.setInitialDelay( 0 );
    timer.setCoalesce(false);



    images = new ImageIcon[numImages];

    new Thread() {
            public void run() {
                loadImages();
            }
    }.start();
}
```

```
public void start() {
    if ( finishedLoading )
        timer.restart();
}


public void loadImages() {
    String prefix = dir + "/flower";
    for ( int i = 0; i < numImages; i++ ) {
        statusLabel.setText( "loading image " + ( i + 1 ) );
        //check if running as an applet or as a standalong application
        if ( inApplet ) {
            try {
              images[i] = new ImageIcon( new URL( getCodeBase() +
                                    prefix + (i+1) + ".jpg" ) );
            } catch( MalformedURLException m ) {
                System.out.println("Couldn't create image: badly formed URL
            }
        }
        else {
            images[i] = new ImageIcon( prefix + (i+1) + ".jpg" );
        }
    }
    finishedLoading = true;
    statusLabel.setText( null );
    timer.start();
}


public void stop() {
    timer.stop();
}


public String getAppletInfo() {
```

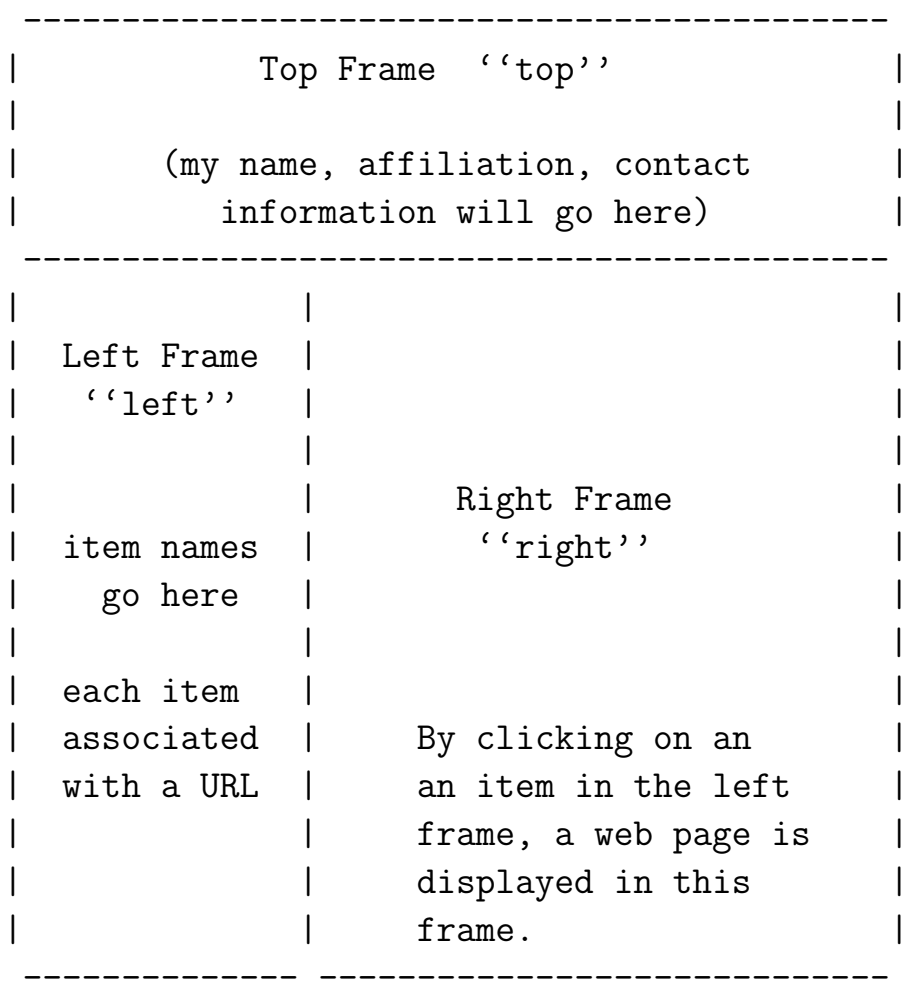
```
        return "Title: A SlideShow Applet\n";
    }

    public String[][] getParameterInfo() {
        String[][] info = {
          {"dir", "String", "the directory containing the images to loop"},
          {"pause", "int", "the time interval between successive frames"},
          {"numImages", "int", "the number of images to display; default is 10
        };
        return info;
    }


    //needed for running program as a stand-alone application
    public static void main( String[] args ) {
        JFrame f = new JFrame( "slide show" );
        f.addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                System.exit(0);
            }
        });

        Container cPane = f.getContentPane();
        SlideShowApplet2 slideshow = new SlideShowApplet2();
        cPane.add( slideshow, BorderLayout.CENTER );
        f.pack();
        f.setSize( 600, 500 );
        f.setVisible( true );
    }
}
```

Applets for Displaying Web Resources:



```
<!doctype html public "html2.0">
<html>
<head>
<title>Experimental HomePage</title>
<meta name="Author" content="Avi Kak">
<meta name="owner"    content="kak@purdue.edu">
<meta name="security" content="public">
</head>
<FRAMESET ROWS="90,*">
<FRAME NAME="top" SRC="top.html" MARGINHEIGHT=2 MARGINWIDTH=2
                                          SCROLLING="no" NORESIZE>
<FRAMESET COLS="180,*">
<FRAME NAME="left" SRC="Left.html" MARGINHEIGHT=2
                            MARGINWIDTH=2 SCROLLING="yes" NORESIZE>
<FRAME NAME="right" SRC="Right.html" MARGINHEIGHT=2
                             MARGINWIDTH=2 SCROLLING="yes" NORESIZE>
</FRAMESET>
</FRAMESET>
</HTML>
```

top.html:

```
<HTML>
<!--- <body bgcolor="#ffffdd">  ----->
<BODY BGCOLOR="#aadddd">
<font size=4>
<B>Avi Kak</B><BR>
<B>Professor of Electrical and Computer Engineering</B><BR>
<B>Purdue University</B>
</font>
<pre>

</pre>
</BODY>
</HTML>
```

left.html:

```
<HTML>
<TITLE>A Trial Applet</TITLE>
<BODY BGCOLOR="#ffffff">
<pre>

</pre>
<font size=2>
<P>
<APPLET CODE="Left.class" WIDTH=180 HEIGHT=480>
<PARAM NAME=item_1 VALUE="Research">
<PARAM NAME=url_1 VALUE=
     "http://RVL4.ecn.purdue.edu/~kak/research-interests.html">
<PARAM NAME=item_2 VALUE="Teaching">
<PARAM NAME=url_2 VALUE=
   "http://RVL4.ecn.purdue.edu/~kak/courses-i-teach/courses-i-teach.html">
<PARAM NAME=item_3 VALUE="Books">
<PARAM NAME=url_3 VALUE="http://RVL4.ecn.purdue.edu/~kak/books.html">
<PARAM NAME=item_4 VALUE="Public Media Pubs">
<PARAM NAME=url_4 VALUE="http://RVL4.ecn.purdue.edu/~kak/other-pubs.html">
<PARAM NAME=item_5 VALUE="CVIU Journal">
<PARAM NAME=url_5 VALUE="http://RVL4.ecn.purdue.edu/~kak/cviu.html">
<PARAM NAME=item_6 VALUE="Robot Vision Lab">
<PARAM NAME=url_6 VALUE="http://rvl1.ecn.purdue.edu/">
</APPLET>
</BODY>
</HTML>
```

```
//Filename: Left.java

import java.awt.*;
import java.awt.event.*;
import java.applet.*;
import java.net.*;
import java.io.*;


public class Left extends Applet implements ActionListener {

  private List links = new List( 10, false );

  public void init()
  {
    setLayout( new BorderLayout() );
    setBackground( Color.red );
    Font f = new Font( "SansSerif", Font.BOLD, 14 );
    setFont( f );

    Panel p = new Panel();
    p.setLayout( new BorderLayout() );
    p.add( links, "Center" );
    links.addActionListener( this );
    int i = 1;
    String s;
    while ( ( s = getParameter( "item_" + i ) ) != null ) {
        links.add( s );
        i++;
    }
    add( p, "Center" );

  }


  public void actionPerformed( ActionEvent evt )
```

```
  {
    try {
      String str = evt.getActionCommand();
      AppletContext context = getAppletContext();

      int i = 1;
      String s;
      while ( ( s = getParameter( "item_" + i ) ) != null ) {
        if ( str.equals( s ) ) {
          URL u = new URL( getParameter( "url_" + i ) );
          context.showDocument( u, "right" );
        }
        i++;
      }
    } catch( Exception e ) { showStatus( "Error " + e ); }
  }
}
```

```
<HTML>
<TITLE>
Web pages will be displayed here
</TITLE>
<BODY BGCOLOR="#dddddd">
double click on any of the items on the left
</BODY>
</HTML>
```

Security Issues Related to Applets:

In general, an applet loaded into a browser cannot read a local file or write into a local file, or execute a local executable file.

An applet is not allowed to even check for the existence of a local file, or rename a local file, or check a file's size, or create a directory on the local disk, etc.

However, when an applet is viewed with the `appletviewer` tool, the applet can be allowed to read files and directories provided those files and directories are named in the `acl.read` property in the `~/.hotjava/properties` file.

An applet viewed through the `appletviewer` tool is also allowed to write into files provided such files are named in `acl.write` property in the `~/.hotjava/properties` file.

```
import java.lang.*;
import java.security.*;

class GetProps {

  public static void main(String[] args) {
    String s;
    try {
      System.out.println("About to get os.name property value");
      s = System.getProperty("os.name", "not specified");
      System.out.println("  The name of your operating system is: " + s);

      System.out.println("About to get java.version property value");
      s = System.getProperty("java.version", "not specified");
      System.out.println("  The version of the JVM you are running is: " +

      System.out.println("About to get user.home property value");
      s = System.getProperty("user.home", "not specified");
      System.out.println("  Your user home directory is: " + s);

      System.out.println("About to get java.home property value");
      s = System.getProperty("java.home", "not specified");
      System.out.println("  Your JRE installation directory is: " + s);
    } catch (Exception e) {
        System.err.println("Caught exception " + e.toString());
    }
  }
}
```

key	meaning
java.version	Java version number
java.vendor	Java vendor-specific string
java.vendor.url	Java vendor URL
java.class.version	Java class version number
os.name	Operating system name
os.arch	Operating system architecture
os.version	Operating system version
file.separator	File separator (eg, "/")
path.separator	Path separator (eg, ":")
line.separator	Line separator

Applets, when viewed either through a web browser or via the `appletviewer`, are prevented by default from reading these system properties:

key	meaning
java.home	Java installation directory
java.class.path	Java classpath
user.name	User account name
user.home	User home directory
user.dir	User's current working directory

However, this default behavior, with regard to both access or not access to the various system properties, can be gotten around for the case when an applet is loaded into an `appletviewer` by suitable changes to the `/.hotjava/properties` file.

Applets, regardless of whether they are loaded into a browser or an `appletviewer`, are not allowed to open network connections to any computer except for the host that supplied the class file for the applet.