



core
WEB
programming

Applets and Basic Graphics

© 2001-2003 Marty Hall, Larry Brown, <http://www.corewebprogramming.com>

Agenda

- Applet restrictions
- Basic applet and HTML template
- The applet life-cycle
- Customizing applets through HTML parameters
- Methods available for graphical operations
- Loading and drawing images
- Controlling image loading
- Java Plug-In and HTML converter

Security Restrictions: Applets Cannot...

- **Read from the local (client) disk**
 - Applets cannot read arbitrary files
 - They can, however, instruct the browser to display pages that are generally accessible on the Web, which might include some local files
- **Write to the local (client) disk**
 - The browser may choose to cache certain files, including some loaded by applets, but this choice is not under direct control of the applet
- **Open network connections other than to the server from which the applet was loaded**
 - This restriction prevents applets from browsing behind network firewalls

Applets Cannot...

- **Link to client-side C code or call programs installed on the browser machine**
 - Ordinary Java applications can invoke locally installed programs (with the `exec` method of the `Runtime` class) as well as link to local C/C++ modules (“native” methods)
 - These actions are prohibited in applets because there is no way to determine whether the operations these local programs perform are safe
- **Discover private information about the user**
 - Applets should not be able to discover the username of the person running them or specific system information such as current users, directory names or listings, system software, and so forth
 - However, applets *can* determine the name of the host they are on; this information is already reported to the HTTP server that delivered the applet

Applet Template

```
import java.applet.Applet;
import java.awt.*;

public class AppletTemplate extends Applet {

    // Variable declarations.

    public void init() {
        // Variable initializations, image loading, etc.
    }

    public void paint(Graphics g) {
        // Drawing operations.
    }
}
```

- Browsers cache applets: in Netscape, use Shift-RELOAD to force loading of new applet. In IE, use Control-RELOAD
- Can use appletviewer for initial testing

5

Applets and Basic Graphics

www.corewebprogramming.com

Applet HTML Template

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
  <TITLE>A Template for Loading Applets</TITLE>
</HEAD>

<BODY>
<H1>A Template for Loading Applets</H1>
<P>
<APPLET CODE="AppletTemplate.class" WIDTH=120 HEIGHT=60>
  <B>Error! You must use a Java-enabled browser.</B>
</APPLET>

</BODY>
</HTML>
```

6

Applets and Basic Graphics

www.corewebprogramming.com

Applet Example

```
import java.applet.Applet;
import java.awt.*;

/** An applet that draws an image. */

public class JavaJump extends Applet {
    private Image jumpingJava; // Instance var declarations here

    public void init() { // Initializations here
        setBackground(Color.white);
        setFont(new Font("SansSerif", Font.BOLD, 18));
        jumpingJava = getImage(getDocumentBase(),
                               "images/Jumping-Java.gif");
        add(new Label("Great Jumping Java!"));
        System.out.println("Yow! I'm jiving with Java.");
    }

    public void paint(Graphics g) { // Drawing here
        g.drawImage(jumpingJava, 0, 50, this);
    }
}
```

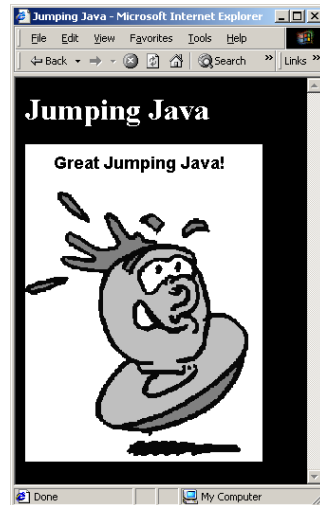
7

Applets and Basic Graphics

www.corewebprogramming.com

Applet Example: Result

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
  <TITLE>Jumping Java</TITLE>
</HEAD>
<BODY BGCOLOR="BLACK" TEXT="WHITE">
<H1>Jumping Java</H1>
<P>
<APPLET CODE="JavaJump.class"
  WIDTH=250
  HEIGHT=335>
  <B>Sorry, this example requires
  Java.</B>
</APPLET>
</BODY>
</HTML>
```



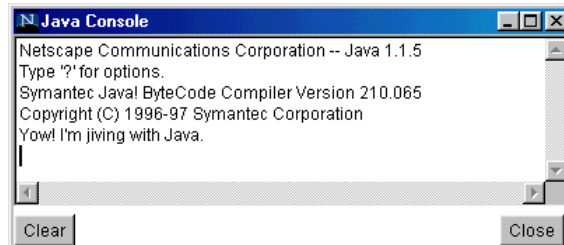
8

Applets and Basic Graphics

www.corewebprogramming.com

Debugging Applets: The Java Console

- **Standard output (from System.out.println) is sent to the Java Console**
 - Navigator: open from Window menu
 - Communicator: open from Communicator ... Tools
 - IE 4: open from View menu (enable from Tools ... Internet Options ... Advanced screen)
 - IE 5/6 with Java plugin: go to Control Panel, click on Java Plugin, and select "Show Console" option.



9

Applets and Basic Graphics

The Applet Life Cycle

- **public void init()**
 - Called when applet is first loaded into the browser.
 - *Not* called each time the applet is executed
- **public void start()**
 - Called immediately after init initially
 - Reinvoked each time user returns to page after having left it
 - Used to start animation threads
- **public void paint(Graphics g)**
 - Called by the browser after init and start
 - Reinvoked whenever the browser redraws the screen (typically when part of the screen has been obscured and then reexposed)
 - This method is where user-level drawing is placed

10

Applets and Basic Graphics

www.corewebprogramming.com

The Applet Life Cycle (Continued)

- **public void stop()**
 - Called when the user leaves the page
 - Used to stop animation threads
- **public void destroy()**
 - Called when applet is killed by the browser
- **Note nonstandard behavior**
 - In some versions of Internet Explorer, and later versions of Netscape, `init` is called each time the user returns to the same page, and `destroy` is called whenever the user leaves the page containing the applet. I.e., applet is started over each time (incorrect behavior!).

11

Applets and Basic Graphics

www.corewebprogramming.com

Useful Applet Methods

- **getCodeBase, getDocumentBase**
 - The URL of the:
 - Applet file** - `getCodeBase`
 - HTML file** - `getDocumentBase`
- **getParameter**
 - Retrieves the value from the associated HTML PARAM element
- **getSize**
 - Returns the `Dimension` (width, height) of the applet
- **getGraphics**
 - Retrieves the current `Graphics` object for the applet
 - The `Graphics` object does not persist across paint invocations

12

Applets and Basic Graphics

www.corewebprogramming.com

Useful Applet Methods (Continued)

- **showDocument (AppletContext method)**

`getAppletContext().showDocument(...)`

- Asks the browser to retrieve and display a Web page
- Can direct page to a named FRAME cell

- **showStatus**

- Displays a string in the status line at the bottom of the browser

- **getCursor, setCursor**

- Defines the `Cursor` for the mouse, for example, `CROSSHAIR_CURSOR`, `HAND_CURSOR`, `WAIT_CURSOR`

Useful Applet Methods (Continued)

- **getAudioClip, play**

- Retrieves an audio file from a remote location and plays it
- JDK 1.1 supports .au only. Java 2 also supports MIDI, .aiff and .wav

- **getBackground, setBackground**

- Gets/sets the background color of the applet
- `SystemColor` class provides access to desktop colors

- **getForeground, setForeground**

- Gets/sets foreground color of applet (default color of drawing operations)

HTML APPLET Element

```
<APPLET CODE="..." WIDTH=xxx HEIGHT=xxx ...>  
...  
</APPLET>
```

- **Required Attributes**

- CODE
 - Designates the filename of the Java class file to load
 - Filename interpreted with respect to directory of current HTML page (default) unless CODEBASE is supplied
- WIDTH and HEIGHT
 - Specifies area the applet will occupy
 - Values can be given in pixels or as a percentage of the browser window (width only). Percentages fail in appletviewer.

15

Applets and Basic Graphics

www.corewebprogramming.com

HTML APPLET Element (Continued)

- **Other Attributes**

- ALIGN, HSPACE, and VSPACE
 - Controls position and border spacing. Exactly the same as with the IMG element
- ARCHIVE
 - Designates JAR file (zip file with .jar extension) containing all classes and images used by applet
 - Save considerable time when downloading multiple class files
- NAME
 - Names the applet for interapplet and JavaScript communication
- MAYSCRIPT (nonstandard)
 - Permits JavaScript to control the applet

16

Applets and Basic Graphics

www.corewebprogramming.com

Setting Applet Parameters

```
<H1>Customizable HelloWWW Applet</H1>

<APPLET CODE="HelloWWW2.class" WIDTH=400 HEIGHT=40>
  <PARAM NAME="BACKGROUND" VALUE="LIGHT">
  <B>Error! You must use a Java-enabled browser.</B>
</APPLET>

<APPLET CODE="HelloWWW2.class" WIDTH=400 HEIGHT=40>
  <PARAM NAME="BACKGROUND" VALUE="DARK">
  <B>Error! You must use a Java-enabled browser.</B>
</APPLET>

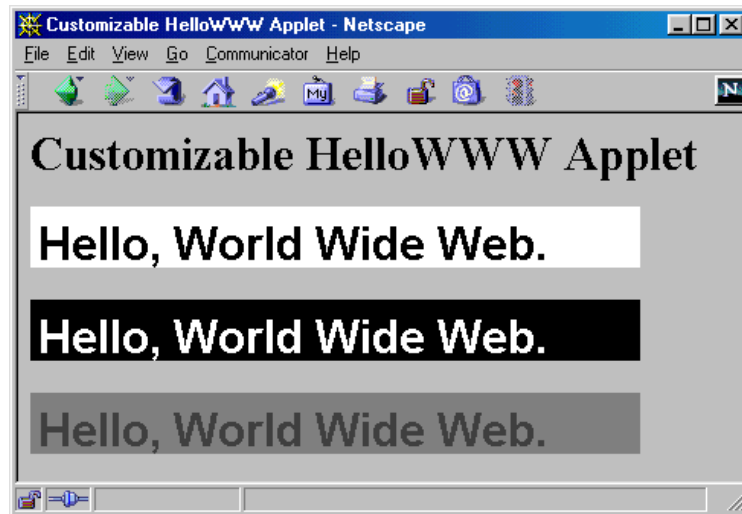
<APPLET CODE="HelloWWW2.class" WIDTH=400 HEIGHT=40>
  <B>Error! You must use a Java-enabled browser.</B>
</APPLET>
```

Reading Applet Parameters

- Use **getParameter(name)** to retrieve the value of the PARAM element
- The name argument is case sensitive

```
public void init() {
    Color background = Color.gray;
    Color foreground = Color.darkGray;
    String backgroundType = getParameter("BACKGROUND");
    if (backgroundType != null) {
        if (backgroundType.equalsIgnoreCase("LIGHT")) {
            background = Color.white;
            foreground = Color.black;
        } else if (backgroundType.equalsIgnoreCase("DARK")) {
            background = Color.black;
            foreground = Color.white;
        }
    }
    ...
}
```

Reading Applet Parameters: Result



19

Applets and Basic Graphics

www.corewebprogramming.com

Useful Graphics Methods

- **drawString(string, left, bottom)**
 - Draws a string in the current font and color with the *bottom left* corner of the string at the specified location
 - One of the few methods where the y coordinate refers to the bottom of shape, not the top. But y values are still with respect to the *top left* corner of the applet window
- **drawRect(left, top, width, height)**
 - Draws the outline of a rectangle (1-pixel border) in the current color
- **fillRect(left, top, width, height)**
 - Draws a solid rectangle in the current color
- **drawLine(x1, y1, x2, y2)**
 - Draws a 1-pixel-thick line from (x1, y1) to (x2, y2)

20

Applets and Basic Graphics

www.corewebprogramming.com

Useful Graphics Methods (Continued)

- **drawOval, fillOval**
 - Draws an outlined and solid oval, where the arguments describe a rectangle that bounds the oval
- **drawPolygon, fillPolygon**
 - Draws an outlined and solid polygon whose points are defined by arrays or a `Polygon` (a class that stores a series of points)
 - By default, polygon is closed; to make an open polygon use the `drawPolyline` method
- **drawImage**
 - Draws an image
 - Images can be in JPEG or GIF (including GIF89A) format

Drawing Color

- **setColor, getColor**
 - Specifies the **foreground color** prior to drawing operation
 - By default, the graphics object receives the foreground color of the window
 - AWT has 16 predefined colors (`Color.red`, `Color.blue`, etc.) or create your own color:
`new Color(r, g, b)`
 - Changing the color of the `Graphics` object affects only the drawing that explicitly uses that `Graphics` object
 - To make permanent changes, call the *applet's* `setForeground` method.

Graphics Font

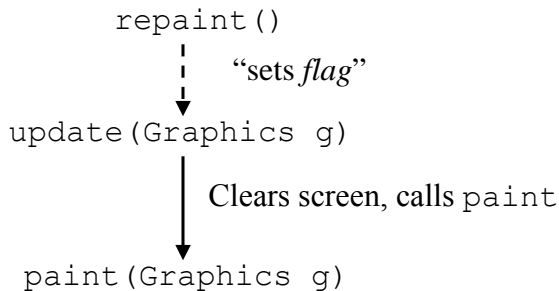
- **setFont, getFont**
 - Specifies the font to be used for drawing text
 - Determine the size of a character through `FontMetrics` (in Java 2 use `LineMetrics`)
 - Setting the font for the `Graphics` object does not persist to subsequent invocations of `paint`
 - Set the font of the window (I.e., call the *applet's* `setFont` method) for permanent changes to the font
 - In JDK 1.1, **only 5 fonts** are available: `Serif` (aka `TimesRoman`), `SansSerif` (aka `Helvetica`), `Monospaced` (aka `Courier`), `Dialog`, and `DialogInput`

Graphic Drawing Modes

- **setXORMode**
 - Specifies a color to XOR with the color of underlying pixel before drawing the new pixel
 - Drawing something **twice in a row** will **restore the original condition**
- **setPaintMode**
 - Set drawing mode back to normal (versus XOR)
 - Subsequent drawing will use the normal foreground color
 - Remember that the `Graphics` object is reset to the default each time. So, no need to call `g.setPaintMode()` in `paint` unless you do non-XOR drawing after your XOR drawing

Graphics Behavior

- **Browser calls `repaint` method to request redrawing of applet**
 - Called when applet first drawn or applet is hidden by another window and then reexposed



Drawing Images

- **Register the Image (from `init`)**

```
Image image = getImage(getCodeBase(), "file");  
Image image = getImage(url);
```

- Loading is done in a separate thread
- If URL is absolute, then `try/catch` block is required

- **Draw the image (from `paint`)**

```
g.drawImage(image, x, y, window);  
g.drawImage(image, x, y, w, h, window);
```

- May draw partial image or nothing at all
- Use the applet (`this`) for the window argument

Loading Applet Image from Relative URL

```
import java.applet.Applet;
import java.awt.*;

/** An applet that loads an image from a relative URL. */

public class JavaMan1 extends Applet {
    private Image javaMan;

    public void init() {
        javaMan = getImage(getCodeBase(),
                           "images/Java-Man.gif");
    }

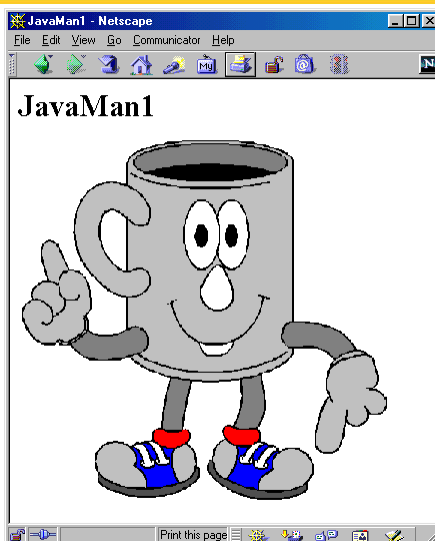
    public void paint(Graphics g) {
        g.drawImage(javaMan, 0, 0, this);
    }
}
```

27

Applets and Basic Graphics

www.corewebprogramming.com

Image Loading Result



28

Applets and Basic Graphics

www.corewebprogramming.com

Loading Applet Image from Absolute URL

```
import java.applet.Applet;
import java.awt.*;
import java.net.*;
...
private Image javaMan;

public void init() {
    try {
        URL imageFile =
            new URL("http://www.corewebprogramming.com" +
                "/images/Java-Man.gif");
        javaMan = getImage(imageFile);
    } catch (MalformedURLException mue) {
        showStatus("Bogus image URL.");
        System.out.println("Bogus URL");
    }
}
```

Loading Images in Applications

```
import java.awt.*;
import javax.swing.*;

class JavaMan3 extends JPanel {
    private Image javaMan;

    public JavaMan3() {
        String imageFile = System.getProperty("user.dir") +
            "/images/Java-Man.gif";
        javaMan = getToolkit().getImage(imageFile);
        setBackground(Color.white);
    }

    public void paintComponent(Graphics g) {
        super.paintComponent(g);
        g.drawImage(javaMan, 0, 0, this);
    }
    ...
}
```

Loading Images in Applications (Continued)

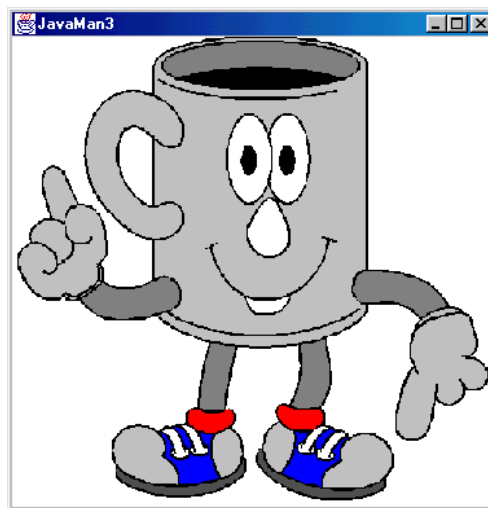
...

```
public void paintComponent(Graphics g) {
    super.paintComponent(g);
    g.drawImage(javaMan, 0, 0, this);
}

public static void main(String[] args) {
    JPanel panel = new JavaMan3();
    WindowUtilities.setNativeLookAndFeel();
    WindowUtilities.openInJFrame(panel, 380, 390);
}
}
```

- See Swing chapter for WindowUtilities

Loading Images in Applications, Result



Controlling Image Loading

- Use `prepareImage` to start loading image

```
prepareImage(image, window)
```

```
prepareImage(image, width, height, window)
```

- Starts loading image immediately (on separate thread), instead of when needed by `drawImage`
- Particularly useful if the images will not be drawn until the user initiates some action such as clicking on a button or choosing a menu option
- Since the applet thread immediately continues execution after the call to `prepareImage`, the image *may* not be completely loaded before `paint` is reached

Controlling Image Loading, Case I: No `prepareImage`

- Image is not loaded over network until **after** `Display Image` is pressed. 30.4 seconds.



Controlling Image Loading, Case 2: With prepareImage

- Image loaded over network immediately. 0.05 seconds after pressing button.



35

Applets and Basic Graphics

www.corewebprogramming.com

Controlling Image Loading: MediaTracker

- Registering images with a MediaTracker to control image loading

```
MediaTracker tracker = new MediaTracker(this);
tracker.addImage(image1, 0);
tracker.addImage(image2, 1);
try {
    tracker.waitForAll();
} catch (InterruptedException ie) {}
if (tracker.isErrorAny()) {
    System.out.println("Error while loading image");
}
```

- Applet thread will block until all images are loaded
- Each image is loaded in parallel on a separate thread

36

Applets and Basic Graphics

www.corewebprogramming.com

Useful MediaTracker Methods

- **addImage**
 - Register a normal or scaled image with a given ID
- **checkAll, checkID**
 - Checks whether all or a particular registered image is done loading
- **isErrorAny, isErrorID**
 - Indicates if any or a particular image encountered an error while loading
- **waitForAll, waitForID**
 - Start loading all images or a particular image
 - Method does not return (blocks) until image is loaded
- **See TrackerUtil in book for simplified usage of MediaTracker**

37

Applets and Basic Graphics

www.corewebprogramming.com

Loading Images, Case I: No MediaTracker

- Image size is wrong, since the image won't be done loading, and -1 will be returned

```
public void init() {
    image = getImage(getDocumentBase(), imageName);

    imageWidth = image.getWidth(this);
    imageHeight = image.getHeight(this);
}

public void paint(Graphics g) {
    g.drawImage(image, 0, 0, this);
    g.drawRect(0, 0, imageWidth, imageHeight);
}
```

38

Applets and Basic Graphics

www.corewebprogramming.com

Loading Images, Case 2: With MediaTracker

- Image is loaded before determining size

```
public void init() {  
    image = getImage(getDocumentBase(), imageName);  
    MediaTracker tracker = new MediaTracker(this);  
    tracker.addImage(image, 0);  
    try {  
        tracker.waitForAll();  
    } catch (InterruptedException ie) {}  
    ...  
    imageWidth = image.getWidth(this);  
    imageHeight = image.getHeight(this);  
}  
  
public void paint(Graphics g) {  
    g.drawImage(image, 0, 0, this);  
    g.drawRect(0, 0, imageWidth, imageHeight);  
}
```

39

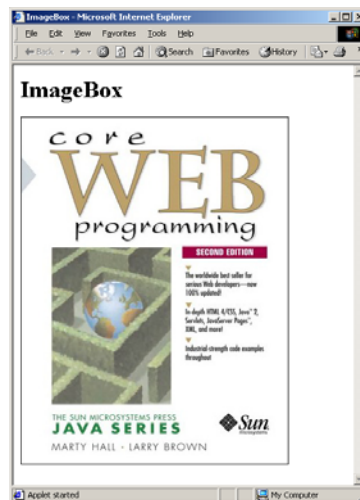
Applets and Basic Graphics

www.corewebprogramming.com

Loading Images: Results



Case 1



Case 2

40

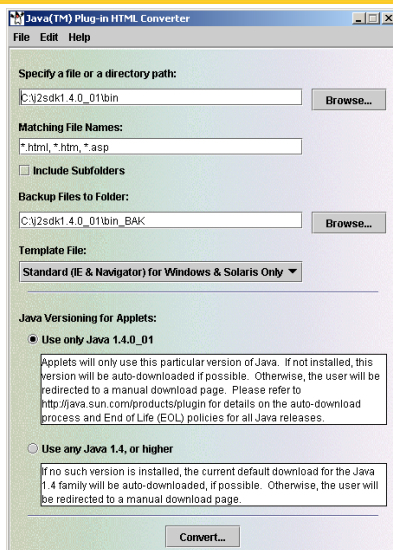
Applets and Basic Graphics

www.corewebprogramming.com

Java Plug-In

- **Internet Explorer and Netscape (except Version 6 and 7) only support JDK 1.1**
- **Plugin provides support for the latest JDK**
 - <http://java.sun.com/products/plugin/>
 - Java 2 Plug-In > 5 Mbytes
 - Installing JDK 1.4 installs plugin automatically
- **Older browsers require modification of APPLET element to support OBJECT element (IE) or EMBED element (Netscape)**
 - Use HTML Converter to perform the modification

Java Plug-In HTML Converter



Java Plug-In HTML Converter

- “Navigator for Windows Only” conversion

```
<EMBED type="application/x-java-applet;version=1.4"
  CODE = "HelloWWW.class" CODEBASE = "applets"
  WIDTH = 400 HEIGHT = 40
  BACKGROUND = "LIGHT"
  scriptable=false
  pluginspage="http://java.sun.com/products/plugin/
    index.html#download"
>
  <NOEMBED>
    <B>Error! You must use a Java-enabled browser.</B>
  </NOEMBED>
</EMBED>
```

Java Plug-In HTML Converter

- “Internet Explorer for Windows and Solaris” conversion

```
<OBJECT classid="clsid:8AD9C840-044E-11D1-B3E9-00805F499D93"
  WIDTH = 400 HEIGHT = 40
  codebase="http://java.sun.com/products/plugin/autodl/
    jinstall-1_4-win.cab#Version=1,4,0,0"
>
  <PARAM NAME = CODE VALUE = "HelloWWW.class" >
  <PARAM NAME = CODEBASE VALUE = "applets" >
  <PARAM NAME="type"
    VALUE="application/x-java-applet;version=1.4">
  <PARAM NAME="scriptable" VALUE="false">
  <PARAM NAME = "BACKGROUND" VALUE ="LIGHT">
  <B>Error! You must use a Java-enabled browser.</B>
</OBJECT>
```

Summary

- **Applet operations are restricted**
 - Applet cannot read/write local files, call local programs, or connect to any host other than the one from which it was loaded
- **The init method**
 - Called only when applet loaded, not each time executed
 - This is where you use `getParameter` to read PARAM data
- **The paint method**
 - Called each time applet is displayed
 - Coordinates in drawing operations are wrt top-left corner
- **Drawing images**
 - `getImage(getCodeBase(), "imageFile")` to “load”
 - `drawImage(image, x, y, this)` to draw

45

Applets and Basic Graphics

www.corewebprogramming.com



core
WEB
programming

Questions?

46

© 2001-2003 Marty Hall, Larry Brown, <http://www.corewebprogramming.com>