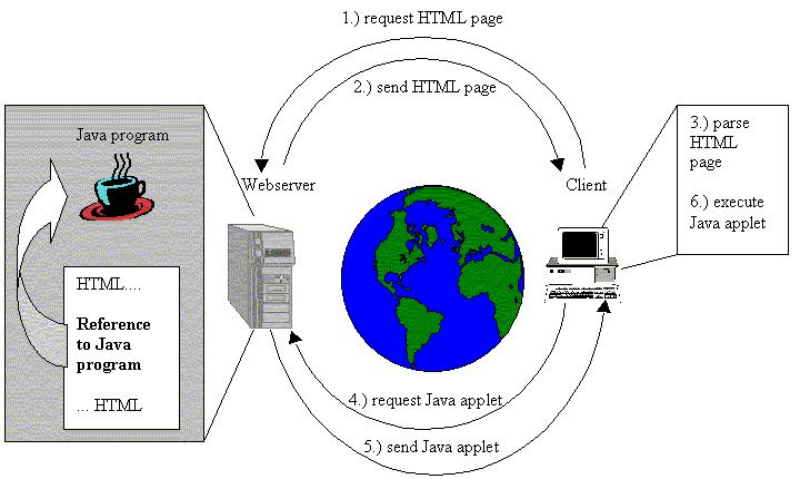
Java Applets

What is an applet?

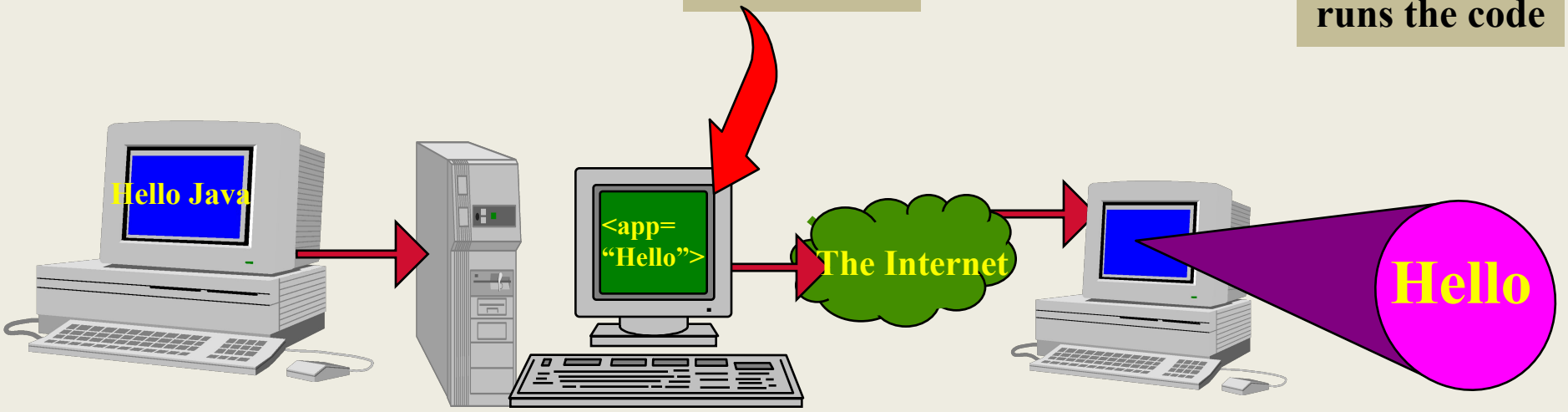
- An **applet** is a small Java program that is typically embedded in a Web page and can be run using the applet viewer or a browser
  - brings web pages to life with interactive content, multimedia, games, and more
  - users can run applets simply by visiting a web page that contains an applet program (if they have the Java runtime environment installed on their computer)



- For security reasons, applets run in a sandbox: they have no access to the client's file system

Applet: Making Web Interactive

1. Applet Development "hello.java" AT SUN.COM
2. hello.class AT SUN'S WEB SERVER
3. Create Applet tag in HTML document
4. Accessing from Your Organisation
5. The browser creates a new window and a new thread and then runs the code

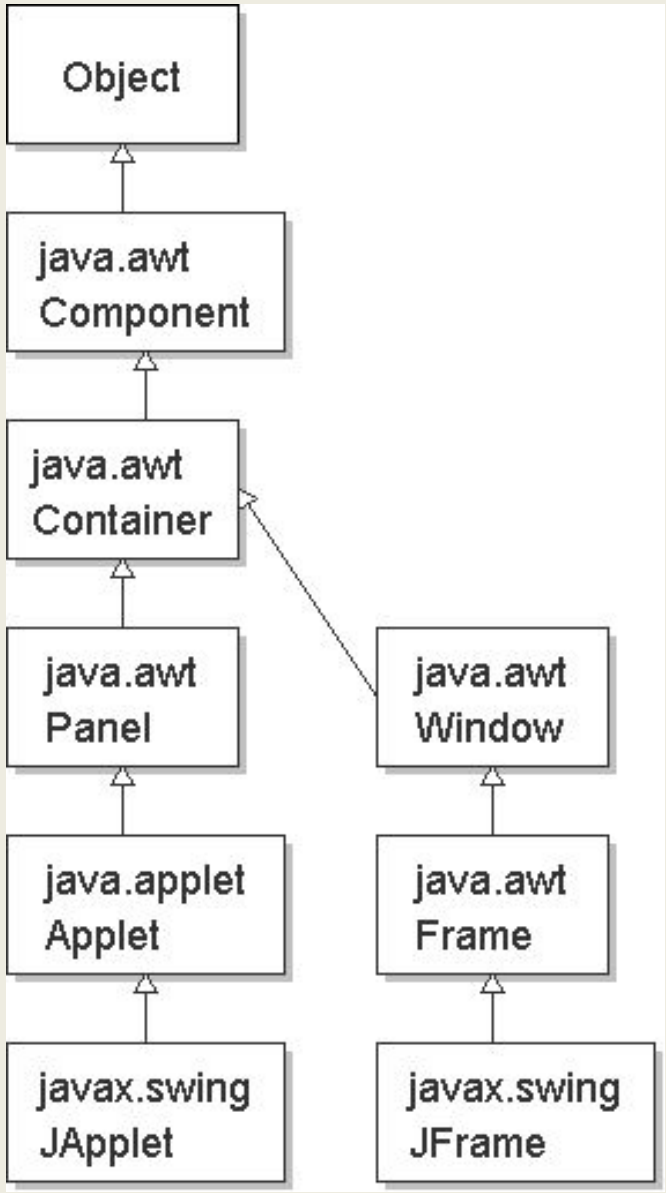


How Applets Differ from Applications

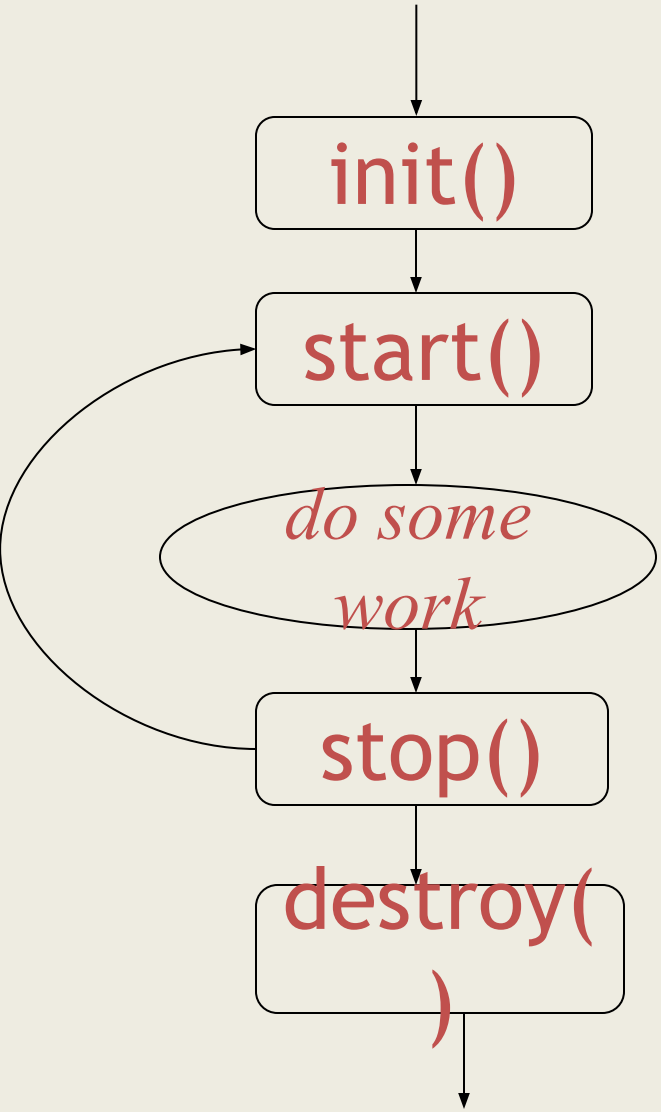
- Although both the Applets and stand-alone applications are Java programs, there are certain restrictions are imposed on Applets due to security concerns:
  - Applets don't use the main() method, but when they are load, automatically call certain methods (init, start, paint, stop, destroy).
  - They are embedded inside a web page and executed in browsers.
  - They cannot read from or write to the files on local computer.
  - They cannot communicate with other servers on the network.
  - They cannot run any programs from the local computer.
  - They are restricted from using libraries from other languages.
- The above restrictions ensures that an Applet cannot do any damage to the local system.

The genealogy of Applet

```
java.lang.Object
   |
   +----java.awt.Component
          |
          +----java.awt.Container
                 |
                 +----java.awt.Panel
                        |
                        +----java.applet.Applet
                               |
                               +----javax.swing.JApplet
```

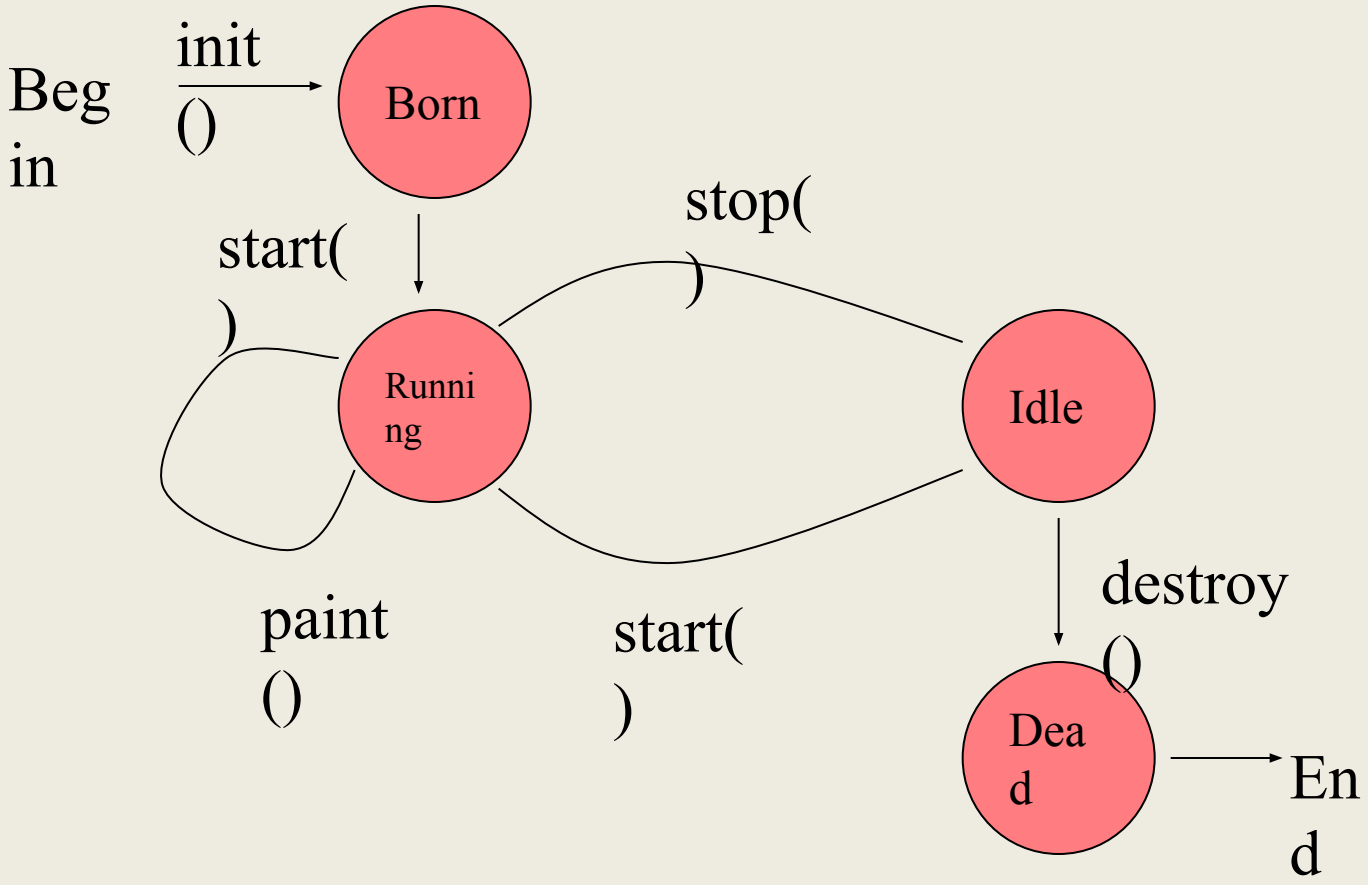


Applet life cycle



- Init() and destroy() are only called once each
- Start() and stop() are called whenever the browser enters and leaves the page
- *do some work* is code called by your *listeners*
- Paint() is called when the applet needs to be repainted

Applet Life Cycle Diagram



Applet methods

public void init ()
public void start ()
public void stop ()
public void destroy ()
public void paint (Graphics)

Also:
public void repaint()
public void update (Graphics)
public void showStatus(String)
public String getParameter(String)

public void init ( )

- init() is the first method to execute
  - init() is an ideal place to initialize variables
  - Almost every applet you ever write will have an init( ) method

start( ), stop( ) and destroy( )

- start() and stop( ) are used when the Applet is doing time-consuming calculations that you don't want to continue when the page is not in front
- public void start() is called:
  - Right after init( )
  - Each time the page is loaded and restarted
- public void stop( ) is called:
  - When the browser leaves the page
  - Just before destroy( )
- public void destroy( ) is called after stop( )
  - Use destroy() to explicitly release system resources (like threads)
  - System resources are usually released automatically

public void paint(Graphics g)

- Needed if you do any drawing or painting other than just using standard GUI Components
- Any painting you want to do should be done here, or in a method you call from here

repaint( )

- Call repaint( ) when you have changed something and want your changes to show up on the screen
  - You *do* need to call repaint() after drawing commands (drawRect(…), fillRect(…), drawString(…), etc.)

- When you call repaint( ), Java schedules a call to update(Graphics ***g***)

update( )

- When you call repaint( ), Java schedules a call to update(Graphics g)
- Here's what update does:

```
public void update(Graphics g) {
   // Fills applet with background color, then
   paint(g);
}
```

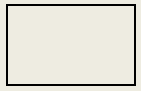
Sample Graphics methods

- A Graphics is something you can paint on

```
g.drawString("Hello", 20, 20);
```

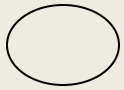
Hello

```
g.drawRect(x, y, width, height);
```



```
g.fillRect(x, y, width, height);
```

```
g.drawOval(x, y, width, height);
```



```
g.fillOval(x, y, width, height);
```

```
g.setColor(Color.red);
```

The simplest possible applet

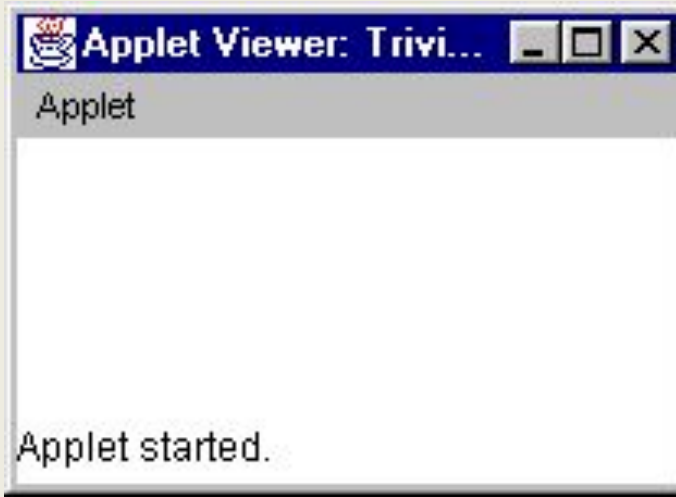
TestApplet.java

```
import java.applet.Applet;

   public class TestApplet extends Applet { }
```

TestApplet.html

```
<applet
    code="TestApplet.class"
    width="150"height="100">
</applet>
```



Your First Java Applet

Hello.java

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

public class Hello extends Applet {
    public void paint(Graphics g) {
        g.drawString("Hello world!", 125, 95);
    }
}
```

hello.html

```
<HTML><BODY>
<APPLET CODE=Hello.class WIDTH=300 HEIGHT=200></APPLET>
</BODY></HTML>
```

- To try it
  - Compile: **`javac Hello.java`**
  - Test: **`appletviewer hello.html`**
  - Or: put all these files in a publicly accessible directory (such as **`~/public_html`** and view using **`netscape`**)
- What happens
  - .html and .class files are slurped over the net
  - The browser has a virtual machine (interpreter) in it
  - It checks for security violations and runs it if ok.

Applet Life Cycle-1/2

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

public class BasicApplet extends Applet {

   public void init() {
System.out.println("init called");
   }
   public void start() {
      System.out.println("start called");
   }
   public void stop(){
      System.out.println("stop  called");
   }
   public void destroy(){
      System.out.println("destroy  called");
   }
```

Applet Life Cycle-2/2

```
   public void paint(Graphics g) {
      System.out.println("paint  called");
      g.drawString("This is basic Applet", 10, 20);}
}
/*
<applet code="BasicApplet" width="300" height="200">
</applet>
*/
```

Passing Parameters to Applets

Param.java

```
public class Param extends Applet
    {
       String str;
       public void init()
       {
                  str = getParameter("string");
              str="hello" + str;
       }
      public void paint (Graphics g)
       {
          g.drawString(str, 10,100);
       }
    }
```

HTML file

```
<HTML>
<HEAD>
<TITLE></TITLE>
<BODY>
    <APPLET CODE = Param.class WIDTH=400 HEIGHT=200>
    <PARAM NAME="string" VALUE "applet">
</APPLET>
</BODY>
</HTML>
```

Using Images

DrawImage.java

```
public class DrawImage extends Applet{
Image image;
   public void init(){
        image=getImage(getDocumentBase(),
getParameter("file"));
   }
   public void paint(Graphics g){
             g.drawImage(image,0,0,this);}
   }


/*<applet code> "drawImage", width=280  height=280>
     < param name="file" value= ".jpg" >
</applet> */
```

JAR Files

- **JAR**: **J**ava **AR**chive. A group of Java classes and supporting files combined into a single file compressed with ZIP format, and given .JAR extension.

1. **create the JAR archive**

- DOS: `jar -cvf filename.jar files`
  Example: `jar -cvf MyAppletJar.jar *.class *.gif *.jpg`

  - some IDEs (JBuilder, Eclipse) can create JARs automatically
    - Eclipse: File -> Export... -> JAR file

2. **Modify your web page to use the JAR file using ARCHIVE attr.**

```
<APPLET code="MyApplet.class" archive="MyAppletJar.jar"
        width=300 height=400> </APPLET>
```