

## Demos

- NPAC Visible Human Viewer
  - <http://zatoka.icm.edu.pl/vh/VisibleHuman.html>
- New York City Subway Map
  - <http://www.brail.org/transit/nycall.html>
- Live Flight tracking
  - <http://live.airportnetwork.com/sfo/>

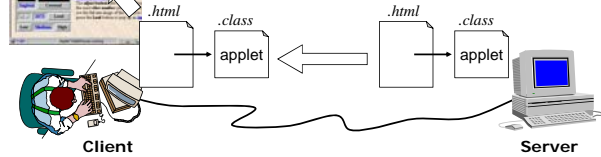
7

## Dynamic Web Pages: HTML + Java

Web browser



- Dynamic pages:
  - Some code is executed (calculations, animations, etc.)
  - Full GUI (buttons, menus, custom drawing, events, etc.)
- The code is executed on the client machine
  - Avoids overloading server CPU
  - Avoids network traffic for exchanging data and results



8

## Applets

- Executed from a web browser – Netscape and Mozilla support latest Java versions, but MS IE only supports JDK 1.1
- Do not use a main method (applets vs. stand-alone applications)
  - The browser invokes some applet methods
- Created by defining a subclass of `java.applet.Applet`

```
class A extends Applet { ... }
```
- The browser creates objects of the applet classes and manages them calling the methods `init`, `start`, `stop`, `destroy`, `paint`
- To program an applet = to override these methods
- Once defined an applet `A`, it can be included in an HTML document with
 

```
<applet code="A.class" width=500 height=20> </applet>
```

  - Path for the `.class` file: `document base`, `codebase` and packages

9

## Applet Methods

- `init()`: initialization, equivalent to a constructor
  - Invoked when the page is loaded (in some browsers, also when going back to the page)
- `start()`: executes applet actions (often creating threads)
  - Invoked when loading a page, when going back to it, or when the browser window is restored after it was minimized
- `stop()`: stop the actions started with `start()`
  - Invoked when the user leaves the page, minimizes or closes the browser
- `destroy()`: additional clean-up after `stop()`
  - Called when the browser is closed
- Inherited methods: `paint`, `update`, `repaint`

10

## Example

HelloWorld.java

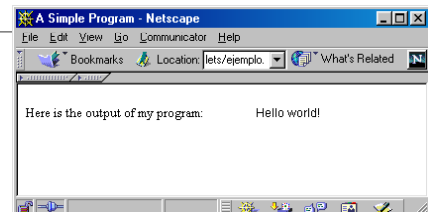
```
import java.applet.Applet;
import java.awt.Graphics;

public class HelloWorld extends Applet {
    public void paint (Graphics g) {
        g.drawString ("Hello world!", 50, 25);
    }
}
```

11

Example.html

```
<HTML>
<HEAD> <TITLE> A Simple Program </TITLE> </HEAD>
<BODY>
    Here is the output of my program:
    <APPLET CODE="HelloWorld.class" WIDTH=150 HEIGHT=25>
    </APPLET>
</BODY>
</HTML>
```



12

## Loading Applets

- The JVM of the browser loads the .class file indicated in <applet...>, retrieving it from the server through the network, if it is not local
- The browser creates an instance of the applet subclass defined in the .class file
- The browser calls the `init()` method on the created object
- The browser calls the `start()` method on the object
- If the applet uses any additional auxiliary class:
  1. The browser checks whether the class is already loaded on the client
  2. If not, it looks for it on the HTML page server
- When the browser window is refreshed, the `paint` method is called on the applet object

13

## Example

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

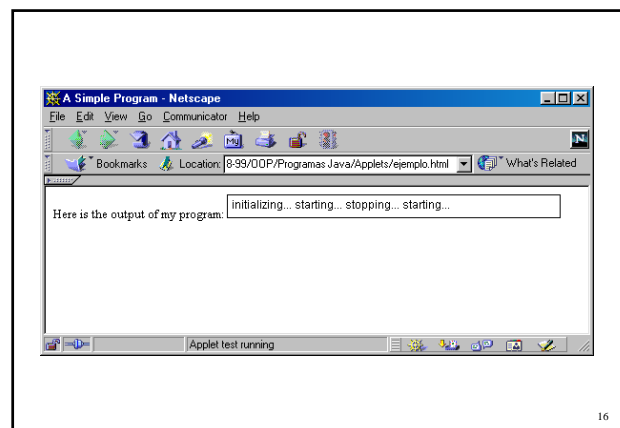
public class Simple extends Applet {
    StringBuffer buffer;
    void addItem (String newWord) {
        System.out.println (newWord);
        buffer.append (newWord);
        repaint ();
    }
    ...
}
```

Standard output:  
Java console of  
the browser

14

```
...
public void init () {
    buffer = new StringBuffer ();
    addItem ("initializing... ");
}
public void start () { addItem ("starting... "); }
public void stop () { addItem ("stopping... "); }
public void destroy () {
    addItem ("preparing for unloading...");
}
public void paint (Graphics g) {
    //Draw a Rectangle around the applet's display area.
    g.drawRect (0, 0, getSize () .width - 1,
               getSize () .height - 1);
    //Draw the current string inside the rectangle.
    g.drawString (buffer.toString (), 5, 15);
}
}
```

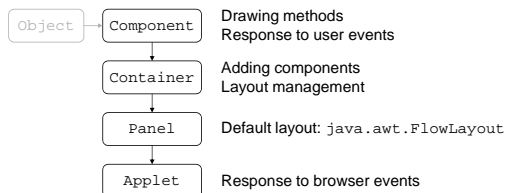
15



16

## An applet is an AWT GUI

- Drawing: `paint`, `update`, `repaint` of `java.awt.Component`
- Add widgets: `add`, `remove`, `setLayout` of `java.awt.Container`
- Response to user input: Implement e.g. `java.awt.event.MouseListener`



- An applet does not require creating a window: it uses that of the browser
- An applet takes a fixed size on the browser that cannot be changed at runtime

17

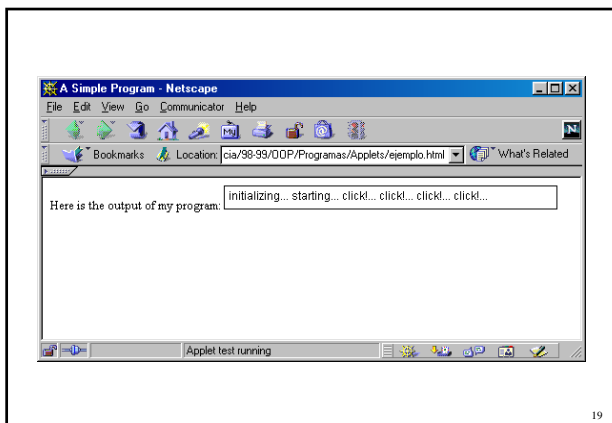
## Response to Events: example

```
public class Click extends Simple implements MouseListener {
    public void init () {
        addMouseListener (this);
        buffer = new StringBuffer ();
        addItem ("initializing... ");
    }
    public void mouseClicked (MouseEvent event) {
        addItem ("click!... ");
    }
    public void mouseEntered (MouseEvent event) { }
    public void mouseExited (MouseEvent event) { }
    public void mousePressed (MouseEvent event) { }
    public void mouseReleased (MouseEvent event) { }
}
```

Method of  
Component

Method of  
MouseListener

18



## Threads and Applets

- An applet may be used from multiple threads
- Many browsers use a separate thread for each applet in an HTML page
- An applet may create several threads in `start()`
- If an applet does a time-consuming task, better do it in the `run()` method of a thread

20

## Threads in Applets: example

```
public class Counter extends Applet implements Runnable {
    boolean running = false; int counter = 0;
    public void run () {
        while (running) {
            repaint ();
            try { Thread.sleep (1000); }
            catch (InterruptedException e) {}
        }
    }
    public void paint (Graphics g) {
        g.drawString (String.valueOf (++counter), 5, 15);
    }
    public void start () {
        if (!running) {
            running = true;
            new Thread (this) .start ();
        }
    }
    public void stop () { running = false; }
}
```

21

## Other Applet Functionalities

- Parameters: similar to parameters of main or constructor
  - `<applet ... > <PARAM NAME=param1 VALUE=value>... <\applet>`
  - `getParameter (String)` method of `Applet`
- Provenance of an applet
  - `getCodeBase ()`
  - `getDocumentBase ()` } → `java.net.URL`
- The `java.applet.AppletContext` interface
  - `Applet: getAppletContext ()` → `AppletContext`
  - `AppletContext: showDocument (URL), showStatus (String), getApplet (String)`
- Sound: the `java.applet.AudioClip` interface
  - `Applet: play (URL), getAudioClip (URL)` → `AudioClip`
  - `AudioClip: play (), loop (), stop ()`

22

## Security!

- Restrictions for remote applets: *the sandbox policy*
  - Cannot read or write on local disk (client)
  - Cannot load local java classes
  - Cannot define native methods or start programs on the client
  - Can only make connections to the server from which they are loaded
- Differences between browsers
- How to live with these restrictions: communicate with a program that runs on the server and does the required operations

23