

Applets

Páginas web estáticas



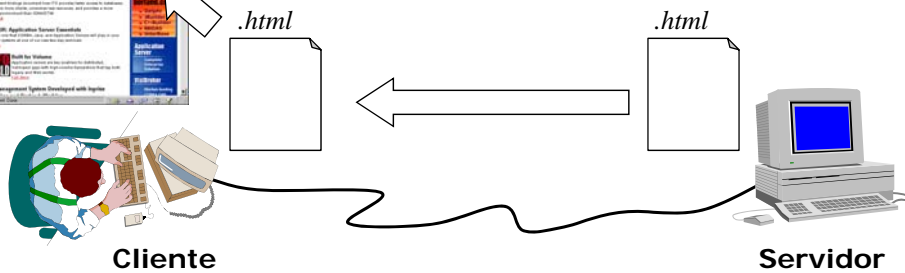
Páginas web estáticas: html

Navegador web



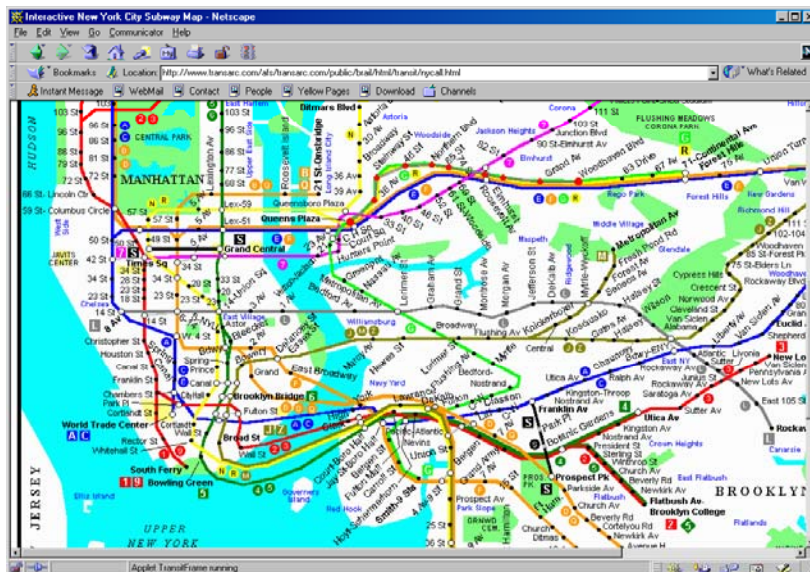
- Texto con formato
- Multimedia: imágenes, sonido, plug-ins
- Hipertexto (links)
- Ejecución de programas en el servidor (CGI's)
- Funcionalidades básicas de interfaz de usuario: botones, campos, etc.

Cientes
sin estado



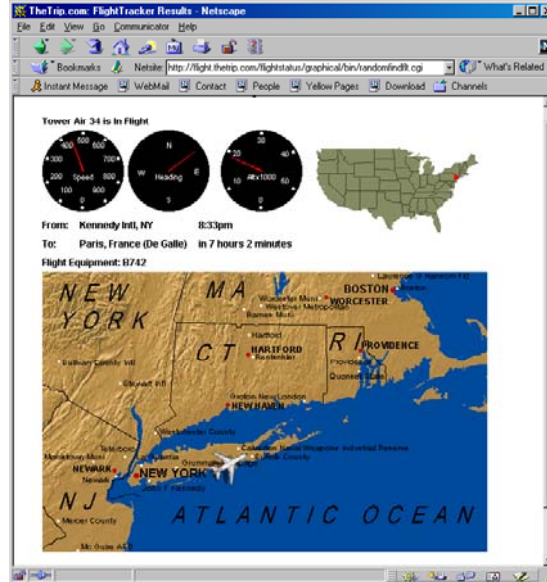
3

Páginas web dinámicas



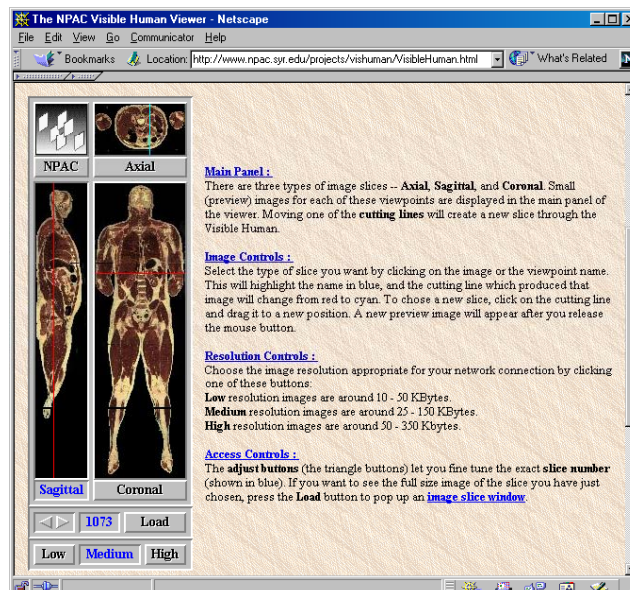
4

Páginas web dinámicas



5

Páginas web dinámicas



6

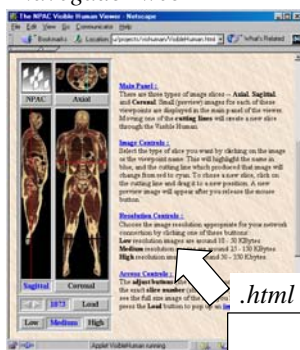
Demos

- 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>
- Molecular visualization
 - <http://openastexviewer.net/web/thinlet.html>
- Chess game
 - <http://english.op.org/~peter/ChessApp/#applet>

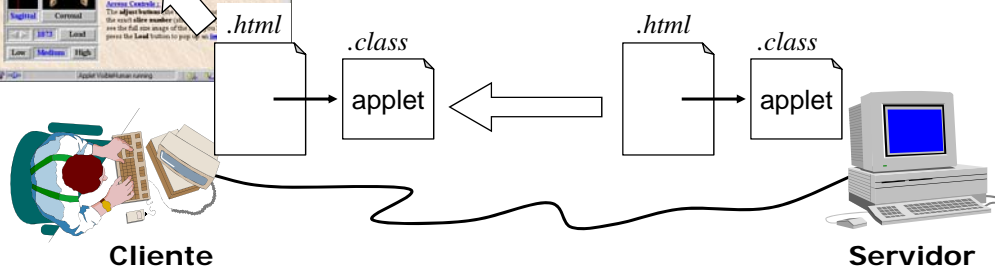
7

Páginas html dinámicas: html + Java

Navegador web



- Páginas dinámicas:
 - Se ejecuta código (cálculos, animaciones, etc.)
 - Interfaz gráfica plena (botones, menús, dibujar, eventos, etc.)
- El código se ejecuta en la máquina cliente
 - Se evita sobrecargar la CPU del servidor
 - Se evita el tráfico de datos y resultados por la red



8

Applets

- Se ejecutan desde un navegador web
- No requieren definir main (applets vs. aplicaciones *stand-alone*)
El navegador es quien invoca los métodos del applet
- Se definen creando una subclase de `java.applet.Applet`

```
class A extends Applet {...} // javax.swing.JApplet
```
- El navegador crea objetos de las clases de applets y los gestiona llamando a los métodos `init`, `start`, `stop`, `destroy`, `paint`
- Programar un applet = sobreescribir estos métodos
- Una vez definido un applet A, se puede incluir en un documento html con

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


Ruta al fichero .class: *document base, codebase* y *packages*

9

Los métodos propios de Applet

- `init()`: inicialización, papel equivalente a un constructor
Invocado al cargar la página (en algunos browsers tb. al cargar de nuevo)
- `start()`: ejecuta las acciones del applet (a menudo iniciando threads)
Invocado al cargar y volver a cargar, o al minimizar + restaurar la ventana
- `stop()`: detener las acciones iniciadas con `start()`
Invocado al abandonar la página, minimizar la ventana o cerrar el navegador
- `destroy()`: clean-up adicional además de `stop()`
Invocado al cerrar el navegador
- Métodos heredados: `paint`, `update`, `repaint`

10

Ejemplo

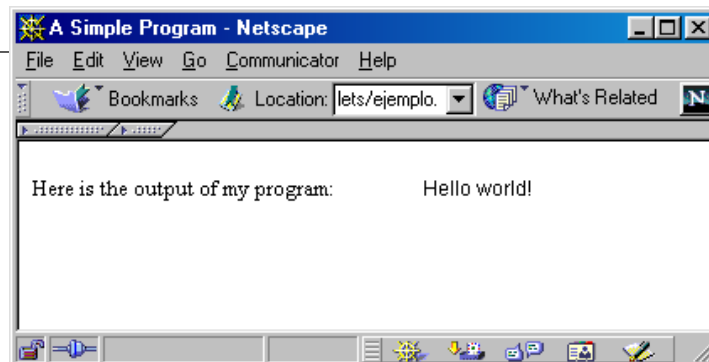
HelloWorld.java

```
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

Ejemplo.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

Carga de un applet

- La JVM del navegador carga el fichero .class indicado en <applet...>, accediendo a él a través de la red si no es local
- El navegador crea una instancia de la subclase de applet definida en el fichero .class
- El navegador invoca el método `init()` sobre el objeto creado
- El navegador invoca el método `start()` sobre el objeto creado
- Si el applet utiliza una clase auxiliar:
 1. El navegador comprueba si la clase ya está cargada en el cliente
 2. Si no lo está, la busca en el servidor de la página html
- Cuando la ventana del navegador se actualiza, invoca al método `paint` del applet

13

Ejemplo

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

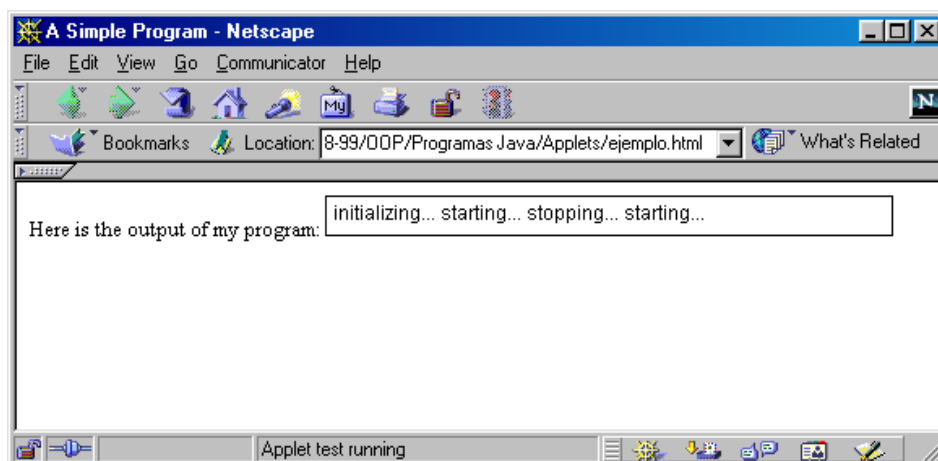
public class Simple extends Applet {
    StringBuffer buffer;
    void addItem (String newWord) {
        System.out.println (newWord);
        buffer.append (newWord);
        repaint ();
    }
    ...
}
```

Standard output:
Java-console
del navegador

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);
}
}
```

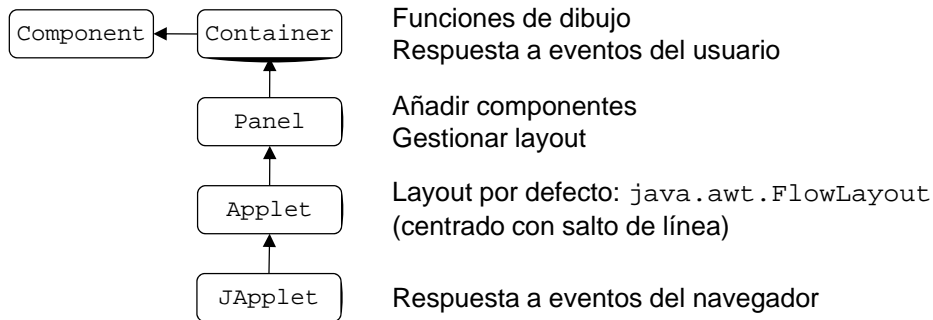
15



16

Un applet es una interfaz gráfica AWT

- Dibujar: `paint`, `update`, `repaint` de `java.awt.Component`
- Añadir widgets: `add`, `remove`, `setLayout` de `java.awt.Container`
- Respuesta a input del usuario:
Implementar p.e. `java.awt.event.MouseListener`



- Un applet no requiere crear una ventana: utiliza la del navegador
- Un applet tiene tamaño fijo, no se puede cambiar

17

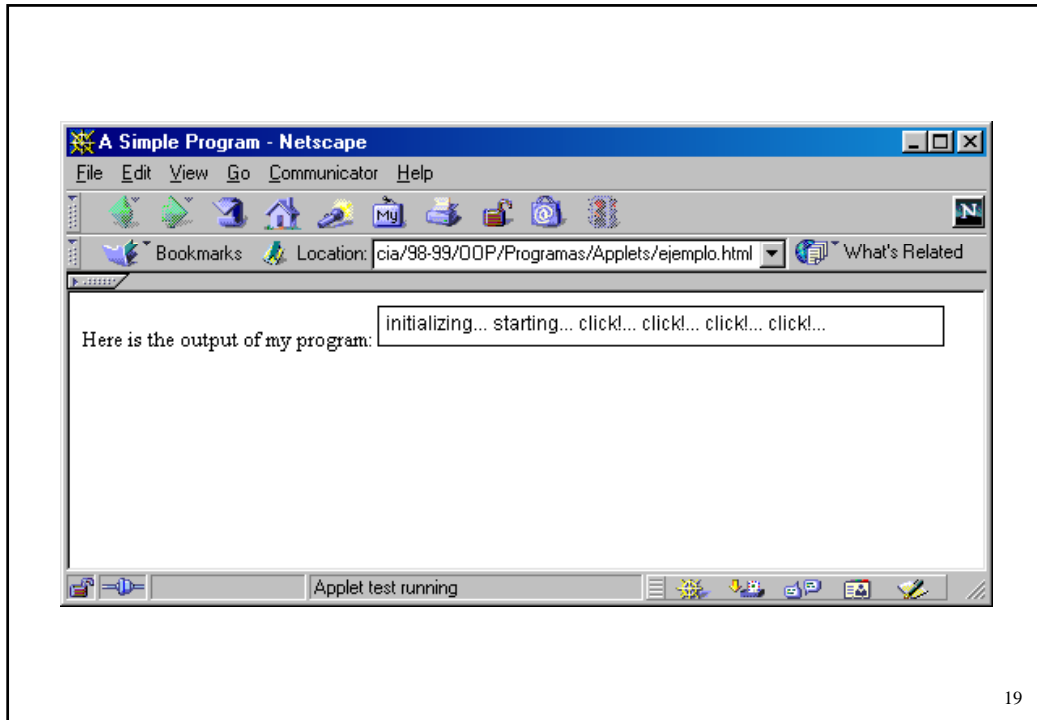
Respuesta a eventos: ejemplo

```
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) { }
}
```

Método de `Component`

Método de `MouseListener`

18



19

Hilos y applets

- Un applet puede pertenecer a múltiples hilos
- Muchos navegadores asignan un hilo para cada applet de una página html
- Un applet puede lanzar varios hilos en `start()`
- Si un applet tiene una tarea que consume mucho tiempo, conviene realizarla en el método `run()` de un hilo

20

Hilos en applets: ejemplo

```
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

Otras funcionalidades de los applets

- Parámetros: papel similar a argumentos de main o de constructor
 - <applet ...> <PARAM NAME=param1 VALUE=valor>... <\applet>
 - Método `getParameter(String)` de `Applet`
- Procedencia del applet
 - `getCodeBase()`
 - `getDocumentBase()` } → `java.net.URL`
- La interfaz `java.applet.AppletContext`
 - `Applet: getAppletContext()` → `AppletContext`
 - `AppletContext: showDocument(URL), showStatus(String), getApplet(String)`
- Sonido: la interfaz `java.applet.AudioClip`
 - `Applet: play(URL), getAudioClip(URL)` → `AudioClip`
 - `AudioClip: play(), loop(), stop()`

22

Seguridad

- Restricciones para los applets remotos: *the sandbox policy*
 - No pueden leer ni escribir en el disco local (cliente)
 - No pueden cargar clases java locales
 - No pueden definir métodos nativos ni arrancar programas en el cliente
 - Sólo pueden realizar conexiones al servidor del que proceden
- Diferencias de unos navegadores a otros
- Para superar las restricciones: comunicarse con un programa que se ejecute en el servidor y realice las funciones deseadas