Apuntes

de

Java

Tema 9:
Applets

Uploaded by

Ingteleco

http://ingteleco.webcindario.com
ingtelecoweb@hotmail.com

La dirección URL puede sufrir modificaciones en el futuro. Si no funciona contacta por email

TEMA 9: APPLETS

9.1.- CONCEPTO DE APPLET

Un applet es una pequeña aplicación Java accesible en un servidor de Internet, que se transporta por la red, se instala automáticamente en nuestra máquina y se ejecuta in situ como parte de un documento web (una página HTML). Un applet es, por tanto, una mínima aplicación Java diseñada para ejecutarse en un navegador Web.

Un applet "vive" incrustado dentro de una página HTML y por tanto, para ejecutar un applet es necesario crear un fichero HTML, incluir el applet dentro de él y cargarlo en un navegador. En consecuencia, los applets no tienen método main() sino que, simplemente, son cargados por el navegador desde un fichero HTML.

Un applet asume en todo momento que su entorno de ejecución es el navegador y su código debe adherirse a una serie de convenciones que le permitan ejecutarse dentro de este entorno. Como consecuencia, todo applet asume una serie de limitaciones que serán expuestas más adelante.

Cada applet está implementado mediante una subclase de la clase Applet. La figura 9.1 muestra la jerarquía de la clase Applet. Esta jerarquía determina en gran medida lo que un applet puede hacer y como lo hace, tal y como se verá mas adelante.

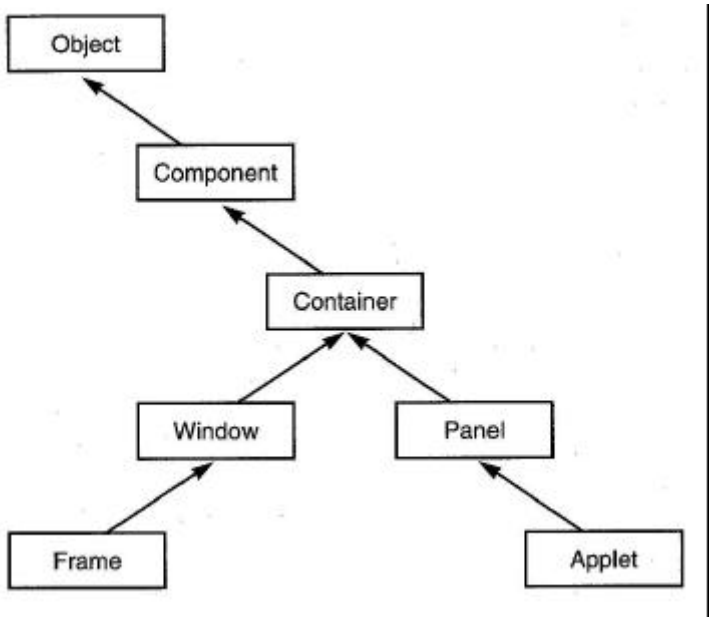

Figura 9.1: Jerarquía de la Clase Applet

El JDK proporciona un visualizador de applets denominado appletviewer.

9.2.- LA ANATOMÍA DE UN APPLET

Para explicar el funcionamiento de los applets y la manera en que se implementan tomaremos como ejemplo el applet ¡Hola Mundo!. Se trata de un applet muy sencillo que escribe el mensaje ¡Hola Mundo!.

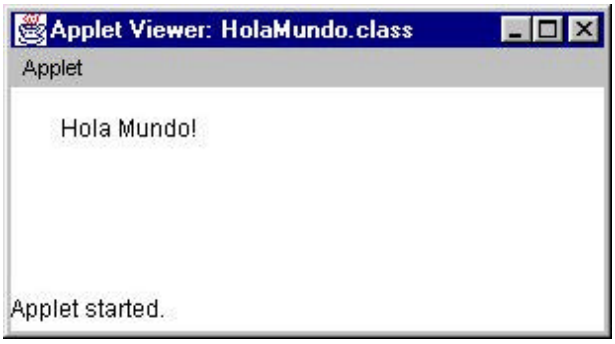

Figura 9.2: Aspecto del applet ¡Hola Mundo!

Este es el código del applet "Hola Mundo" que explicaremos detalladamente en los apartados posteriores:

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

public class HolaMundo extends Applet {
    public void paint(Graphics g) {
        g.drawString("Hola Mundo!", 50, 25);
    }
}
```

9.2.1.- Importar Clases y Paquetes

Las dos primeras líneas del siguiente listado importan dos clases utilizadas en el applet: Applet y Graphics.

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

public class HolaMundo extends Applet {
    public void paint(Graphics g) {
        g.drawString("Hola Mundo!", 50, 25);
    }
}
```

Los dos paquetes en los que se encuentran estas clases, java.applet y java.awt, son parte del corazón del API de Java - API con el que cada programa Java puede contar dentro del entorno Java.

El paquete java.applet contiene clases que son esenciales para los apples Java. El paquete java.awt contiene las clases más utilizadas en la herramienta de Ventanas Abstractas (AWT) que proporciona el interface gráfico de usuario (GUI) de Java.

9.2.2.- Heredar de la Clase Applet

La primera línea en negrita del siguiente listado empieza un bloque que define la clase HolaMundo.

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

public class HolaMundo extends Applet {
    public void paint(Graphics g) {
        g.drawString("Hola Mundo!", 50, 25);
    }
}
```

Para crear un applet creamos una nueva clase, en este caso HolaMundo, que extienda de la clase de Java: Applet. De esta forma, heredamos toda la funcionalidad necesaria para crear nuestro applet.

De la clase Applet, los applets heredan gran cantidad de funcionalidades. Quizás la más importante es la habilidad de responder a las peticiones del navegador. Por ejemplo, cuando un navegador compatible con Java carga una página que contiene un applet, el navegador envía una petición al applet, para que éste se inicialice y empiece su ejecución. Así, modificando determinados métodos heredados de la clase Applet podremos conseguir que nuestro applet lleve a cabo las funciones que deseamos.

Un applet no está restringido a definir sólo una clase. Junto con la necesaria subclase Applet, un applet puede definir clases de usuario adicionales. Cuando un applet intenta ejecutar una clase, la apliación busca la clase en el ordenador local. Si la clase no está disponible localmente, la carga desde la posición donde fuera originaria la subclase Applet.

9.2.3.- Implementar Métodos en un Applet

Las líneas en negrita del siguiente listado implementan el método paint().

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

public class HolaMundo extends Applet {
    public void paint(Graphics g) {
        g.drawString("Hola Mundo!", 50, 25);
    }
}
```

Todos los applets deben implementar uno o más de estos métodos: init(), start(), o paint(). Junto con los métodos init(), start(), y paint(), un applet puede implementar dos métodos más que el navegador puede llamar cuando ocurre un evento principal (como salir de la página del applet): stop() y destroy(). Todos estos métodos son métodos heredados de la clase Applet y se explicarán con más detalle en secciones posteriores.

Por tanto, los applets pueden:

- Sobrescribir algunos de los métodos que han heredado de la superclase Applet.

- Implementar nuevos métodos que sean necesarios para proporcionar la funcionalidad requerida.

En nuestro applet de todos los métodos heredados de la superclase Applet, únicamente vamos a sobrescribir el método paint(). En la clase Applet se llama al método paint() cada vez que el applet arranca o necesita ser refrescado. De acuerdo a las normas de redefinición de métodos, ahora cuando se invoque el método paint(), se ejecutará este último paint() y no el paint() de la clase Applet (que está vacío).

Volviendo al código anterior, el objeto Graphics pasado dentro del método paint() representa el contexto de dibujo en la pantalla del applet. El primer argumento del método drawString() es la cadena que se muestra en la pantalla. El segundo argumento son las posiciones (x,y) de la esquina inferior izquierda del texto en la pantalla. Este applet muestra la cadena "Hola Mundo" en la posición (50,25). Las coordenadas de la ventana del applet empiezan en (0,0), que es la esquina superior izquierda de la ventana del applet.

Para dibujar en un applet, ya sea un string, una línea, una figura, etc, hay que utilizar un objeto de la clase Graphics. Otros métodos, por ejemplo, que tiene la clase Graphics, son:

```
drawLine( int x1,int y1,int x2,int y2 )
drawRect( int x,int y,int ancho,int alto )
drawOval( int x,int y,int ancho,int alto )
```

9.2.4.- Compilación de Applets

Ahora que tenemos el código de nuestro applet básico necesitamos compilarlo y obtener un fichero .class ejecutable. Se utiliza el compilador Java del JDK, javac, para realizar la tarea. El comando de compilación será:

```
c:>javac HolaMundo.java
```

Eso es todo. El compilador javac generará un fichero HolaMundo.class que podrá ser llamado desde cualquier navegador con soporte Java y, por tanto, capaz de ejecutar applets Java.

9.2.5.- Crear un Fichero HTML que incluya un Applet

Como ya se ha comentado, todo applet debe incrustarse dentro de una página HTML para poder ser ejecutado. Para ello, vamos a crear un fichero llamado HolaMundo.html en el mismo directorio que contiene HolaMundo.class. Este fichero HTML deberá contener el siguiente texto:

```
<HTML>
<HEAD>
<TITLE> A Programa Sencillo </TITLE>
</HEAD>
<BODY>
Aquí está la salida de mi programa:
<APPLET CODE="HolaMundo.class" WIDTH=150 HEIGHT=25>
</APPLET>
</BODY>
</HTML>
```

Las líneas en negrita del listado anterior comprenden la etiqueta <APPLET> que incluye el applet "Hola Mundo" en una página HTML.

La etiqueta <APPLET> especifica que el navegador debe cargar la clase cuyo código compilado está en el fichero llamado HolaMundo.class. El navegador busca este fichero en el mismo directorio del documento HTML que contiene la etiqueta. Cuando el navegador encuentra el fichero de la clase, carga el bytecode a través de la red (si es necesario) hasta el ordenador donde se está ejecutando el navegador. Después, el navegador crea un ejemplar de la clase.

Los atributos WIDTH y HEIGHT tienen el mismo significado que en otras etiquetas HTML (por ejemplo, la etiqueta <IMG>): Especifican el tamaño en pixels del área de la pantalla reservado para el applet.

Nota: Para ejecutar los applets de Java es necesario usar un navegador que los soporte. Esto es debido a que la etiqueta <APPLET> no está incluida en el estándar de HTML y por tanto no todos los navegadores tienen porqué comprenderla. En cualquier caso, los navegadores más importantes (Microsoft Internet Explorer y Netscape Navigator) la soportan.

En apartados posteriores se proporcionará más información sobre la inclusión de Applets en páginas HTML y sobre el uso de la etiqueta <APPLET>.

9.2.6.- Ejecución de un Applet

El JDK, incluye el visor de applets básico, appletviewer, que puede utilizarse para la visualización rápida y prueba de nuestros applets. La ejecución de un applet sobre appletviewer se realiza a través de la llamada:

```
c:> appletviewer HolaMundo.html
```

En nuestro caso el fichero con el código HTML que ejecutará nuestro applet HolaMundo es HolaMundo.html que generará la salida que se ve en la figura 9.3:

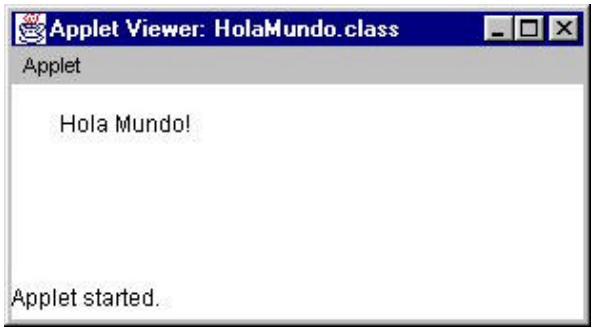

Figura 9.3: Ejecución del applet ¡Hola Mundo!

9.3.- EJEMPLO 1: APPLET QUE USA FUENTES Y COLORES

Veamos ahora cómo construir un applet que escribe un string utilizando fuentes (tipos de letras) y colores. El resultado de la ejecución de este applet se muestra en la figura 9.4.

Figura 9.4: Applet que usa fuentes y colores

Su código es el siguiente:

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

public class AppletHolaColor extends Applet {
Font tipoFuente= new Font("TimesRoman", Font.BOLD, 25);
     public void paint (Graphics g)
{
          setBackground(Color.white);
          g.setFont(tipoFuente);
          g.setColor(Color.blue);
          g.drawString("Hola a todos en colores", 100, 50);
     }
}
```

9.4.- CICLO DE VIDA DE UN APPLET

A continuación se enumeran los pasos que se llevan a cabo durante la carga de un applet por parte de un navegador.

- El navegador comienza a interpretar la página HTML donde se encuentra insertado el applet. Cuando encuentra la etiqueta <APPLET> el navegador se descarga del mismo servidor donde se encontraba la página HTML el bytecode de la clase indicada en la etiqueta <APPLET>. En este caso, HolaMundo.class.
- Se crea una instancia de la clase que controla el applet. En el ejemplo de la figura anterior, sería una instancia de la clase HolaMundo.
- El applet se inicializa (ejecución del método init()).
- El applet comienza a ejecutarse (ejecución del método start()).

- El applet se visualiza en pantalla (ejecución del método paint()).

La figura 9.5 resume los pasos anteriores.

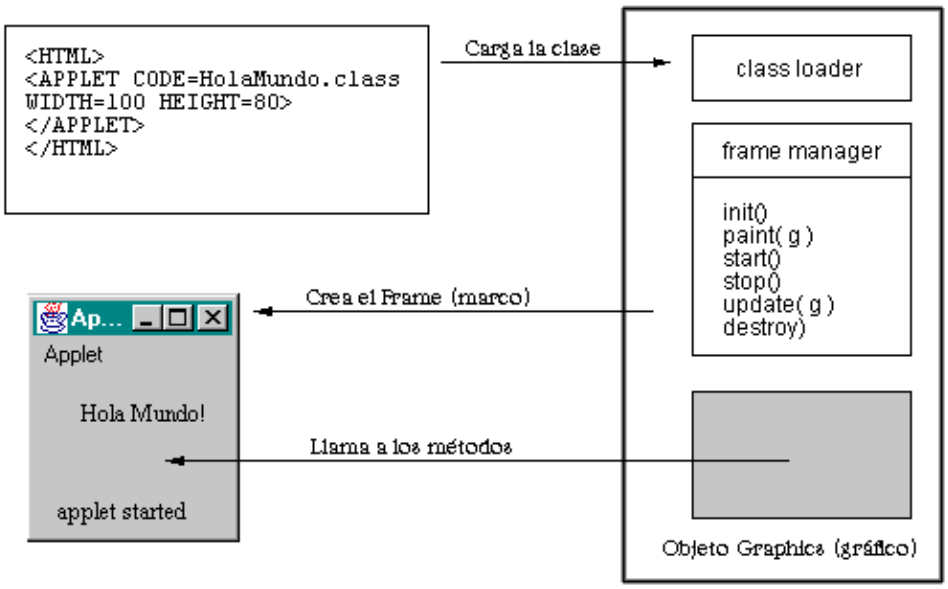

Figura 9.5: Ciclo de vida de un applet

- Cuando se abandona la página, por ejemplo, para visitar un enlace, el applet detiene la ejecución. Cuando se regresa a la página que contiene el applet, se reanuda la ejecución.

- Si se utiliza la opción del navegador de Reload, es decir, volver a cargar la página, el applet es descargado y vuelto a cargar. El applet libera todos los recursos que hubiese acaparado, detiene su ejecución y ejecuta su finalizador para realizar un proceso de limpieza final de sus trazas. Después de esto, el applet se descarga de la memoria y vuelve a cargarse volviendo a comenzar su inicialización.

- Finalmente, cuando se concluye la ejecución del navegador, o de la aplicación que está visualizando el applet, se detiene la ejecución del applet y se libera toda la memoria y recursos ocupados por el applet antes de salir del navegador.

- Durante todo el tiempo de vida del applet, éste continuará recibiendo llamadas de métodos por parte del navegador que le irán informando de los distintos eventos que se vayan produciendo (por ejemplo, maximizar y minimizar el navegador, refresco de la pantalla, abandono de la página HTML,...). Estas llamadas pueden ser recibidas asíncronamente y serán tratadas con más detalle en el siguiente apartado.

9.5.- MÉTODOS DE LOS APPLETS

Incluso para el applet más sencillo necesitaremos varios métodos. Son los que se usan para arrancar (start) y detener (stop) la ejecución del applet, para pintar (paint) y actualizar (update) la pantalla y para capturar la información que se pasa al applet desde el fichero HTML a través de la marca APPLET.

A continuación se explicará la utilidad de cada uno de ellos:

init()

Esta función miembro es llamada al crearse el applet. Es llamada sólo una vez. La clase padre Applet no hace nada en init(). Las clases derivadas deben redefinir este método para cambiar el tamaño durante su inicialización y para realizar cualquier otra inicialización de datos que solamente deba realizarse una vez. En general, el método init() debería contener el código que se pondría normalmente en un constructor.

Deberían realizarse al menos las siguientes acciones:

- Carga de imágenes y sonido.
- El resize del applet para que tenga su tamaño correcto.
- Asignación de valores a las variables globales.

Por ejemplo:

```
public void init() {
if( width < 200  ||  height < 200 )
     resize( 200,200 );
     valor_global1 = 0;
     valor_global2 = 100;

     // cargaremos imágenes en memoria sin mostrarlas cargaremos música de
     // fondo en memoria sin reproducirla
}
```

start()

Llamada para activar el applet. Esta función miembro es llamada cuando el applet se hace visible al usuario (cuando, por ejemplo, se maximiza la ventana del navegador donde se estaba ejecutando el applet o se vuelve a la página tras haberla abandonado momentáneamente). La clase Applet no hace nada en este método. Las clases derivadas deberían redefinirlo para comenzar una animación, sonido, etc.

```
public void start() {
     estaDetenido = false;

     // comenzar la reproducción de la música
     musicClip.play();
}
```

stop()

Llamada para detener el applet. Se llama cuando el applet desaparece de la pantalla (cuando, por ejemplo, minimizamos la ventana del navegador o abandonamos la página para, a través de una enlace, visitar otra). La clase Applet no hace nada en este método. Las clases derivadas deberían redefinirlo para detener la animación, el sonido, etc.

```
public void stop() {
```

```
        estaDetenido = true;

        if( /* ¿se está reproduciendo música? */ )
            musicClip.stop();
    }
```

La mayoría de los applets que sobrescriben el método start() también deberían sobrescribir el método stop(). Este método debería parar la ejecución del applet, para que no gaste recursos del sistema cuando el usuario no esta viendo la página del applet. Por ejemplo, un applet que muestra animaciones debe parar de mostrarlas cuando el usuario no esta mirando.

destroy()

Esta función miembro es llamada cuando el applet no se va a usar más. La clase Applet no hace nada en este método. Las clases derivadas deberían redefinirlo para hacer una limpieza final de todos los recursos utilizados. Los applet multithread deberán usar destroy() para "matar" cualquier thread del applet que quedase activo.

La mayoría de los applets no necesitan sobrescribir el método destroy(), porque su método stop() (al que se llama antes del método destroy()) hace todo lo necesario para detener la ejecución del applet. Sin embargo, el método destroy() esta disponible para los applets que necesitan liberar recursos adicionales.

resize(int width, int height)

El método init() debería llamar a esta función miembro para establecer el tamaño del applet. Puede utilizar las variables ancho y alto, pero no es necesario. Cambiar el tamaño en otro sitio que no sea init() produce un reformateo de todo el documento y no se recomienda.

En el navegador Netscape, el tamaño del applet es el que se indica en la marca APPLET del HTML y no hace caso a lo que se indique desde el código Java del applet.

width

Variable entera, su valor es el ancho definido en el parámetro WIDTH de la marca HTML del APPLET. Por defecto es el ancho del icono.

height

Variable entera, su valor es la altura definida en el parámetro HEIGHT de la marca HTML del APPLET. Por defecto es la altura del icono. Tanto width como height están siempre disponibles para que se puede chequear el tamaño del applet.

Podemos retomar el ejemplo de init():

```
public void init() {
    if( width < 200  ||  height < 200 )
        resize( 200,200 );
        ...
```

paint(Graphics g)

Se llama cada vez que se necesita refrescar el área de dibujo del applet. La clase Applet simplemente dibuja una caja con sombreado de tres dimensiones en el área. Obviamente, la clase derivada debería redefinir este método para representar algo en la pantalla.

Podemos utilizar paint() para imprimir nuestro mensaje de bienvenida:

```
void public paint( Graphics g ) {
     g.drawString( "Hola Java!",25,25 );
     // Dibujaremos la imágenes que necesitemos
}
```

update(Graphics g)

Esta es la función que se llama realmente cuando se necesita actualizar la pantalla (en lugar de llamar directamente a la función paint()). La clase Applet simplemente limpia el área y llama al método paint(). Esta funcionalidad es suficiente en la mayoría de los casos. De cualquier forma, las clases derivadas pueden sustituir esta funcionalidad para sus propósitos.

repaint()

A esta función se la debería llamar cuando el applet necesite ser repintado. No debería sobrecargarse, sino dejar que Java repinte completamente el contenido del applet.

Al llamar a repaint(), sin parámetros, internamente se llama a update() que borrará el rectángulo sobre el que se redibujará y luego a su vez llamará a paint().

Este es el método que habitualmente invocan los programadores de applets para refrescar el contenido del applet en pantalla. Al ser una función sin argumentos, es más fácil de invocar que los correspondientes métodos paint() y update() que requieren como parámetro el objeto Graphics.

getParameter(String attr)

Este método carga los valores pasados al applet a través de la marca APPLET de HTML. El argumento String es el nombre del parámetro que se quiere obtener. Devuelve el valor que se le haya asignado al parámetro; en caso de que no se le haya asignado ninguno, devolverá null.

Para usar getParameter(), se define una cadena genérica. Una vez que se ha capturado el parámetro, se utilizan métodos de cadena o de números para convertir el valor obtenido al tipo adecuado.

```
public void init()
{
     String pv;

     pv = getParameter( "velocidad" );
     if( pv == null )
          velocidad = 10;
```

```
        else
            velocidad = Integer.parseInt( pv );
    }
```

getDocumentBase()

Indica la ruta http, o el directorio del disco, de donde se ha recogido la página HTML que contiene el applet, es decir, el lugar donde está el documento HTML en la red o en el disco local.

getCodeBase()

Indica la ruta http, o el directorio del disco, de donde se ha cargado el código bytecode que forma el applet, es decir, el lugar donde está el fichero .class en la red o en el disco.

print(Graphics g)

Para imprimir en impresora, al igual que paint() se puede utilizar print(), que pintará en la impresora el mapa de bits del dibujo.

Es importante darse cuenta de que muchos de estos métodos son en realidad métodos pertenecientes a la clase Component de la librería AWT, esto es así porque la clase Applet (de la que extienden todos los applets) extiende de la clase Panel (que a su vez extiende de Container y a su vez de Component) y por tanto son métodos heredados por todos los applets (ver la jerarquía de la figura 9.1).

No todos los applets necesitan sobrescribir todos estos métodos. Algunos applets muy sencillos sobrescriben algunos de ellos o ninguno, por ejemplo el "Applet Hola Mundo" únicamente sobrescribe el método paint(), ya que no hace nada excepto dibujarse a si mismo. Sin embargo, la mayoría de los applets, hacen mucho más.

9.6.- EJEMPLO 2: CICLO DE VIDA DE UN APPLET

A continuación se presenta el código de un applet con el que se podrá experimentar los eventos que van aconteciendo durante el ciclo de vida del mismo. Este applet se llama Sencillo y redefine los 4 métodos principales del ciclo de vida de los applets (init, start, stop y destroy), mostrando en cada uno de ellos un mensaje indicativo del evento que ha ocurrido.

Este es su código:

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

public class Simple extends Applet {
    StringBuffer buffer;
    public void init() {
        buffer = new StringBuffer();
        addItem("inicializando... ");
    }
```

```
    public void start() {
        addItem("arrancando... ");
    }

    public void stop() {
        addItem("parando... ");
    }

    public void destroy() {
        addItem("preparando para descargar...");
    }

    void addItem(String newWord) {
        System.out.println(newWord);
        buffer.append(newWord);
        repaint();
    }

    public void paint(Graphics g) {
        //Dibuja un Rectangulo alrededor del area del Applet.
        g.drawRect(0, 0, size().width - 1, size().height - 1);

        //Dibuja la cadena actual dentro del Rectangulo.
        g.drawString(buffer.toString(), 5, 15);
    }
}
```

9.7.- INSERCIÓN DE UN APPLET EN UNA PÁGINA HTML

Para insertar un applet en una página HTML necesitamos utilizar la etiqueta <APPLET>, cuya sintaxis se muestra a continuación:

```
<APPLET CODE=  WIDTH=  HEIGHT=  [CODEBASE=] [ALT=] [NAME=]
   [ALIGN=] [VSPACE=] [HSPACE=]>
</APPLET>
```

Como se puede ver, hay algunos atributos que son obligatorios y otros opcionales. Todos los atributos, siguiendo la sintaxis de HTML, se especifican de forma: atributo=valor.

Los **atributos obligatorios** son los siguientes:

- **CODE:** Nombre de la clase principal.

  Indica el fichero de clase ejecutable, que tiene la extensión .class.

- **WIDTH:** Anchura inicial.

  Indica la anchura inicial que el navegador debe reservar para el applet en pixels.

- **HEIGHT:** Altura inicial.

Indica la altura inicial en pixels. Un applet que disponga de una geometría fija no se verá redimensionado por estos atributos. Por ello, si los atributos definen una zona menor que la que el applet utiliza, únicamente se verá parte del mismo, como si se visualiza a través de una ventana, eso sí, sin ningún tipo de desplazamiento.

Los **atributos opcionales** son:

- **CODEBASE:** URL base del applet.

  Se emplea para utilizar el URL base del applet. En caso de no especificarse, se utilizará el mismo que tiene el documento.

- **ALT:** Texto alternativo.

  Muestra un texto alternativo al applet. Esto sirve para navegadores en modo texto o que entiendan la etiqueta APPLET pero no implementen la máquina virtual Java.

- **NAME:** Nombre de la instancia.

  Otorga un nombre simbólico a esta instancia del applet en la página, que puede ser empleado por otros applets de la misma página para poder ser localizado. Así, un applet puede ser cargado varias veces en la misma página tomando un nombre simbólico distinto en cada momento.

- **ALIGN:** Justificación del applet.

  Se emplea para alinear el applet permitiendo al texto fluir a su alrededor. Puede tomar los siguientes valores: LEFT, RIGHT, TOP, TEXTTOP, MIDDLE, ABSMIDDLE, BASELINE, BOTTOM y ABSBOTTOM.

- **VSPACE:** Espaciado vertical.

  Indica el espaciado vertical entre el applet y el texto, en pixels. Sólo funciona cuando se ha indicado ALIGN=LEFT ó RIGHT.

- **HSPACE:** Espaciado horizontal.

  Funciona igual que el anterior, pero indicando espaciamiento horizontal, en pixels. Sólo funciona cuando se ha indicado ALIGN=LEFT ó RIGHT.

Es probable encontrar en algunas distribuciones otras etiquetas para la inclusión de applets, como <APP>. Esto se debe a que estamos ante la tercera revisión de la extensión de HTML para la incrustación de applets y ha sido adoptada como la definitiva. Por ello, cualquier otro medio corresponde a implementaciones obsoletas que han quedado descartadas.

La marca ALT del Applet la utilizaría un navegador que entendiese la marca APPLET, pero que por alguna razón, no pudiese ejecutarlo.

ALT no es utilizado por los navegadores que no entienden la marca APPLET, por ello se ha definido la marca </APPLET>, que finaliza la descripción del applet. Un navegador con

soporte Java ignorará todo el texto que haya entre las dos marcas <APPLET> y </APPLET>, sin embargo, un navegador que no soporte Java ignorará las marcas y presentará el texto que se encuentre entre ellas.

9.7.1.- ¿Dónde se coloca la Página HTML? ¿Y el Applet?

A la hora de ejecutar un applet es necesario que la página HTML que contiene al applet se encuentre situada en una determinada posición (dentro del sistema de directorios de nuestra máquina) con respecto al fichero .class donde se encuentra el applet.

La figura 9.6 muestra la posición que debe ocupar toda página HTML que contenga a un applet, en función de si éste pertenece a un determinado paquete o no.

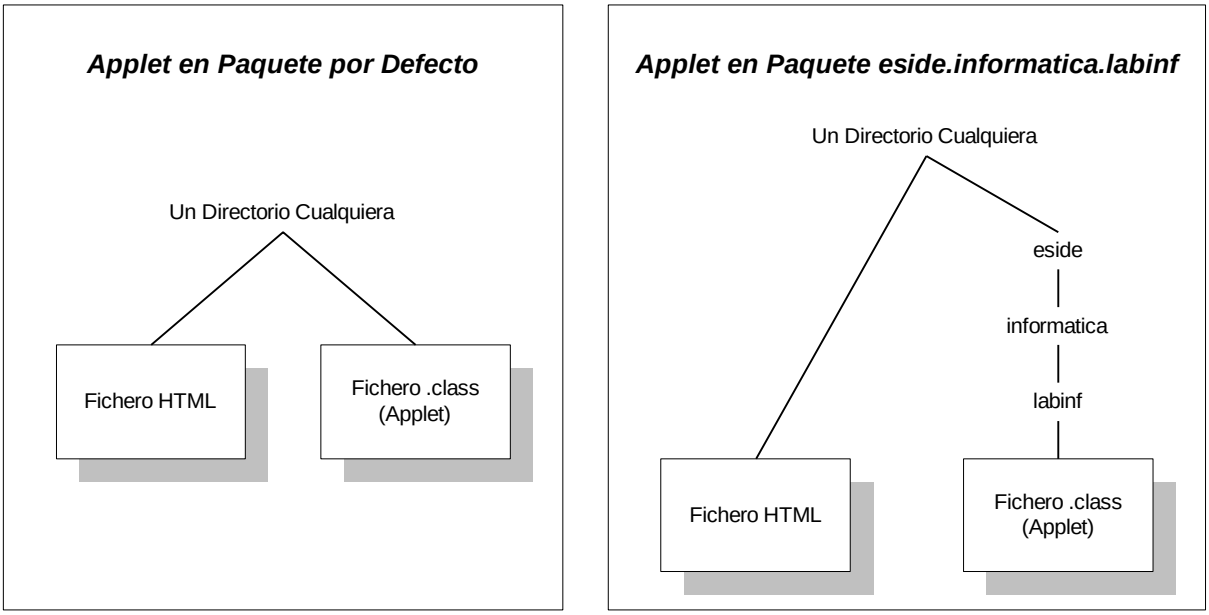

Figura 9.6: Ubicación del Applet y de la página HTML

Tal y como se muestra en la figura, si el applet no se encuentra dentro de ningún paquete (se encontrará, por tanto, dentro del paquete por defecto), tanto la página HTML como el applet (fichero .class) deberán situarse dentro del mismo directorio.

En cambio, si el applet se encuentra dentro de algún paquete (en este caso el paquete eside.informatica.labinf), la página HTML deberá situarse justo antes del directorio donde comienza la estructura de directorios correspondiente al paquete del applet (en este caso la estructura eside\informatica\labinf).

9.8.- EJEMPLO 3: APPLET QUE MUESTRA UNA FOTO

Vamos a construir un applet que muestra una imagen. El resultado de la ejecución de este applet se muestra en la figura 9.7.

Figura 9.7: Applet que muestra una imagen

Su código es el siguiente:

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

public class AppletFoto extends Applet
{
     private Image foto;

     // Método init() se llama cada vez que el visor carga el applet.
     // Sólo se ejecuta 1 vez (no como el método start())
     public void init()
     {
          //getImage() devuelve un objeto Image
          // Parámetros:
          //   getDocumentBase() devuelve la URL del documento donde el
          //   applet está inmerso.
          //   String donde se almacena el nombre de un fichero GIF o JPEG
          foto=getImage(getDocumentBase(), "../Imagenes/Foto1.jpg");
     }

     public void paint (Graphics g) {
          // drawImage() tiene como parámetros
          //   - objeto Image
          //   - Coordenadas de la esquina superior izquierda
          //   - Un objeto ImageObserver, es donde se muestra la imagen
          g.drawImage(foto,0,0,this);
     }
}
```

Este applet sólo mostraría esa foto. Realmente esto se podría hacer sin utilizar un applet, metiendo la foto en la página directamente. Normalmente a los applets se les dará más funcionalidad.

Para insertar una imagen en una página HTML, se utiliza la etiqueta <IMG>. Vamos a ver cómo sería un documento HTML que incluya nuestro applet, pero además, incluya otras dos fotos.

```
<HTML>
<HEAD>
<TITLE> Visor de fotos </TITLE>
</HEAD>
<BODY>

<H1 align=center> Ejemplo de applet </H1>

<P>Applet que muestra una imagen
<BR>
<APPLET CODE="AppletFoto.class" WIDTH=320 HEIGHT=200>
</APPLET>
<hr>
<IMG SRC="../Imagenes/Foto2.gif">
Ejemplo de imagen GIF en HTML

<hr>
<IMG SRC="../Imagenes/Foto3.jpg">
Ejemplo de imagen JPEG en HTML

</BODY>
</HTML>
```

9.9.- PASO DE PARÁMETROS A LOS APPLETS

El espacio que queda entre las marcas de apertura y cierre de la definición de un applet, se utiliza para el paso de parámetros al applet. Para ello se utiliza la marca PARAM en la página HTML para indicar los parámetros, con la siguiente sintaxis:

```
<APPLET CODE=  WIDTH=  HEIGTH= ...>
<PARAM NAME=  VALUE= >
</APPLET>
```

En el código interno del applet se pueden recoger los parámetros así pasados con el método getParameter() de la clase java.applet.Applet. La construcción puede repetirse cuantas veces se quiera, una tras otra.

Los atributos que acompañan a la marca PARAM son los siguientes:

- **NAME:** Nombre del parámetro que se desea pasar al applet.
- **VALUE:** Valor que se desea transmitir en el parámetro que se ha indicado antes.
- **Texto HTML:** Texto HTML que será interpretado por los navegadores que no entienden la marca APPLET en sustitución del applet mismo.

Para mostrar esta posibilidad podemos modificar el applet básico HolaMundo para que pueda saludar a alguien. Tenemos que pasarle al applet como parámetro el nombre de la persona que vaya a saludar. Para ello, creamos la clase HolaNombre.java:

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

public class HolaNombre extends Applet
{
String nombre;

     public void init() {
     nombre = getParameter( "Nombre" );
     }

     public void paint( Graphics g ) {
     g.drawString( "¡Hola " + nombre + "!", 25, 25 );
     }
}
```

Si compilamos el ejemplo obtendremos el fichero HolaNombre.class que incluiremos en nuestra página Web. Vamos a generar el fichero HolaNombre.html, en el que incluiremos nuestro applet, y que debería tener el siguiente contenido:

```
<HTML>
<APPLET CODE=HolaNombre.class WIDTH=300 HEIGHT=100>
<PARAM NAME="Nombre" VALUE="Jaimito">
</APPLET>
</HTML>
```

Si se quiere cambiar el nombre de la persona a la que debe saludar, no haría falta modificar el código Java, ni será necesario recompilarlo. Sólo habría que cambiar el valor del parámetro "Nombre" en la página HTML.

Los parámetros no se limitan a uno solo. Se puede pasar al applet cualquier número de parámetros y siempre hay que indicar un nombre y un valor para cada uno de ellos.

El método getParameter() es fácil de entender. El único argumento que necesita es el nombre del parámetro cuyo valor queremos recuperar. Todos los parámetros se pasan como Strings, en caso de necesitar pasarle al applet un valor entero, se ha de pasar como String, recuperarlo como tal y luego convertirlo al tipo que deseemos. Tanto el argumento de NAME como el de VALUE deben ir colocados entre comillas dobles, (") ya que son String.

El hecho de que las marcas <APPLET> y <PARAM> sean ignoradas por los navegadores que no entienden Java, es inteligentemente aprovechado a la hora de definir un contenido alternativo a ser mostrado en este último caso. Así la etiqueta es doble:

```
<APPLET atributos>
parámetros
contenido alternativo
</APPLET>
```

Nuestro fichero, para mostrar el applet de ejemplo, lo modificaremos para que pueda ser visualizado en cualquier navegador y en unos casos presente la información alternativa y en otros, ejecute nuestro applet:

```
<HTML>
<APPLET CODE=HolaNombre.class WIDTH=300 HEIGHT=100>
<PARAM NAME="Nombre" VALUE="Jaimito">
No verás lo bueno hasta que consigas un navegador
<I>Java Compatible</I>
</APPLET>
</HTML>
```

9.10.- EJEMPLO 4: PASO DE PARÁMETROS A APPLETS

Construcción de un applet que recibe una foto como parámetro, además escribe sobre la foto el autor y el nombre del fichero que contiene la foto. El resultado de la ejecución de este applet se muestra en la figura 9.8.

Figura 9.8: Applet que recibe una foto como parámetro

El **código de este applet** es el siguiente:

```
import java.applet.Applet;
import java.awt.Color;
import java.awt.Font;
import java.awt.Graphics;
import java.awt.Image;

public class AppletFotoParametro extends Applet
{
     private Image foto;
     Font tipoFuente=new Font("TimesRoman", Font.BOLD, 12);
     String parametro1;

     //   método init() se llama cada vez que el visor carga el applet
     public void init()
     {
```

```
            // Toma un parámetro de donde está inmerso el Applet
            parametro1 = getParameter("Foto");

            //getImage() devuelve un objeto Image
            // parámetros:
            //   - getDocumentBase() devuelve la URL del documento donde el
            //                       applet está inmerso
            //   - String donde se alamacena el nombre de un fichero GIF o JPEG
            foto=getImage(getDocumentBase(),parametro1);
      }

      public void paint( Graphics g ) {
            g.setFont( tipoFuente );
            g.setColor( Color.blue );
            g.translate( getInsets().left, getInsets().top );

            // drawImage() tiene como parámetros
            //   - objeto Image
            //   - Coordenadas de la esquina superior izquierda
            //   - Un objeto ImageObserver, es donde se muestra la imagen
            g.drawImage(foto,0,0,this);
            g.drawString("Foto JMCL - "+parametro1,15,15);
      }
}
```

Fichero HTML en el que utilizaríamos el applet:

```
<HTML>
<HEAD>
<TITLE> Visor de fotos </TITLE>
</HEAD>
<BODY>

<H1 align=center> Ejemplo de applet con parámetros </H1>

<P>Applet que muestra una imagen que se pasa como parametro
<BR>
<APPLET   CODE="AppletFotoParametro.class" WIDTH=320 HEIGHT=200>
<PARAM NAME="Foto" VALUE="../Imagenes/Bayas.jpg">
</APPLET>

<APPLET   CODE="AppletFotoParametro.class" WIDTH=320 HEIGHT=200>
<PARAM NAME="Foto" VALUE="../Imagenes/Pijiba10.gif">
</APPLET>

<APPLET   CODE="AppletFotoParametro.class" WIDTH=320 HEIGHT=200>
<PARAM NAME="Foto" VALUE="../Imagenes/Baleira.jpg">
</APPLET>

<APPLET   CODE="AppletFotoParametro.class" WIDTH=320 HEIGHT=200>
<PARAM NAME="Foto" VALUE="../Imagenes/Tamesis.jpg">
</APPLET>

</BODY>
</HTML>
```

En esta página estamos sacando 4 applets, cada uno mostrando una foto diferente. Con el appletviewer, veríamos así todos los applets:

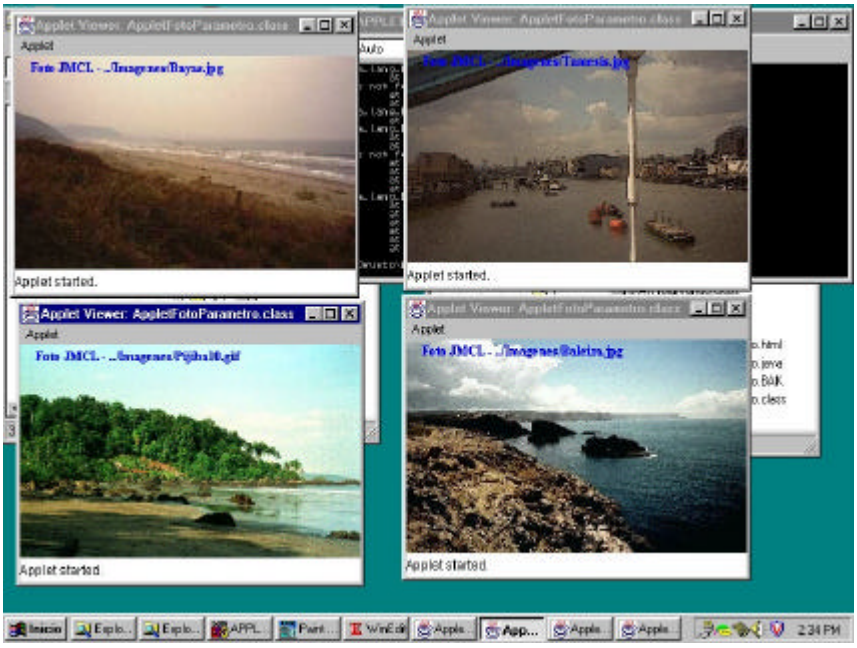

Figura 9.9: Ejecución de una página con 4 applet con el appletviewer

9.11.- QUÉ PUEDE Y QUÉ NO PUEDE HACER UN APPLET

Esta página ofrece una amplia perspectiva tanto de las restricciones de los applets, como de las características especiales que éstos tienen.

9.11.1.- Restricciones de Seguridad

Cada navegador implementa unos controladores de seguridad para asegurarse de que los applets no hagan ningún daño. Esta sección describe los controladores de Seguridad que poseen los navegadores actuales. Sin embargo, la implementación de controladores de seguridad es diferente de un navegador a otro. Los controladores de Seguridad también están sujetos a cambios. Por ejemplo, si el navegador está desarrollado para trabajar sólo en entornos fiables, entonces sus controladores de seguridad serán mucho más restrictivos que los descritos en esta sección

Los Navegadores actuales imponen las siguientes restricciones a los applets que se cargan a través de la Red:

- Un applet no puede cargar librerías ni definir métodos nativos.
- No puede leer ni escribir ficheros en el Host en el que se está ejecutando.
- No puede realizar conexiones en la Red, excepto con el Host del que fue cargado.
- No puede arrancar ningún programa en el Host donde se está ejecutando.

- No puede leer ciertas propiedades del sistema.
- Las ventanas que proporcionan los applets tienen un aspecto diferente a las de cualquier aplicación.

9.11.2.- Capacidades de los Applets

El paquete java.applet proporciona una API que contiene algunas capacidades de los applets que las aplicaciones no tienen. Por ejemplo, los applets pueden ejecutar sonidos en un navegador, mientras otros programas no pueden.

Aquí tienes algunas cosas curiosas que pueden hacer los applets:

- Los Applets pueden hacer conexiones al host del que fueron cargados.
- Los Applets que se ejecutan dentro de un navegador Web pueden hacer que se muestren páginas HTML de una forma muy sencilla.
- Los Applets pueden invocar métodos públicos de otros Applets que se encuentren en la misma página.
- Los Applets que se han cargado desde un directorio local (desde un directorio en el CLASSPATH del usuario) no tienen ninguna restricción como los applets cargados a través de la Red.
- Aunque la mayoría de los applets paran su ejecución cuando el usuario abandona la página, no tienen porque hacerlo.

9.12.- SUMARIO

Este apartado resume todo lo que se ha aprendido, añadiendo un poco más de información para ayudar a adquirir una visión integrada de todo lo tratado anteriormente.

Lo primero que se aprendió para escribir un applet es que se debe crear una subclase de la clase java.applet. En esta subclase, se debe implementar al menos uno de los siguientes métodos: init(), start(), y paint(). Los métodos init() y start(), junto con stop() y destroy(), son los eventos más importantes que ocurren en el ciclo de vida de un applet. Se llama al método paint() cuando el applet necesita dibujarse en la pantalla.

La clase Applet desciende de la clase AWT Panel, que desciende a su vez de la clase AWT Container, que desciende a su vez de la clase AWT Component. De la clase Component, un applet hereda las capacidades de dibujar y manejar eventos. De la clase Container, un applet hereda la capacidad de añadir otros componentes y de tener un manejador de distribución para controlar su tamaño y posición. De la clase Panel, un applet hereda mucho, incluyendo la capacidad de responder a los principales eventos en el ciclo de vida, como son la carga y la descarga.

Los applets se incluyen en páginas HTML utilizando la etiqueta <APPLET>. Cuando un usuario visita una página que contiene un applet, esto es lo que sucede:

1. El navegador encuentra el fichero .class que contiene la subclase Applet. La posición del fichero .class (que contiene los bytecodes Java) se especifica con el atributo CODEBASE de la etiqueta <APPLET>.

2. El navegador trae los bytecodes a través de la red al ordenador del usuario.

3. El navegador crea un ejemplar de la subclase Applet.

4. El navegador llama al método init() del applet. Este método realiza la inicialización.

5. El navegador llama al método start() del applet. Éste método normalmente arranca las tareas del applet.

Lo principal en un applet es la subclase Applet, clase controladora, pero los applets también pueden utilizar otras clases. Estas otras clases pueden ser propias del navegador, proporcionadas como parte del entorno Java o clases del usuario suministradas por ti. Cuando un applet intenta ejecutar una clase por primera vez, el navegador intenta encontrarla en el host donde se está ejecutando el navegador. Si no puede encontrarla allí, la busca en el mismo lugar de donde cargó la subclase Applet del applet. Cuando el navegador encuentra la clase, carga sus bytecodes (a través de la red, si es necesario) y continua la ejecución del applet.

Cargar código ejecutable a través de la red es un riesgo de seguridad. Para los applets Java, algunos de estos riesgos se reducen, porque el lenguaje Java está diseñado para ser seguro -por ejemplo, no permite punteros a posiciones de memoria. Además, los navegadores compatibles con Java proporcionan seguridad imponiendo algunas restricciones. Estas restricciones incluyen no permitir a los applets la carga de código escrito en otro lenguaje que no sea Java, y tampoco permiten leer o escribir ficheros en el host donde se está ejecutando el navegador.

9.13.- EJERCICIOS

9.13.1.- Applet que muestra la fecha y la hora

Construir un applet (junto con su página HTML) que se comporte como un reloj, mostrando continuamente la fecha y la hora actual en el formato que se muestra en la figura 9.10.

Figura 9.10: Applet que muestra la fecha y la hora

Aunque en la figura no puede apreciarse, el applet debe modificar continuamente (segundo a segundo) la fecha y la hora visualizada de tal modo que siempre visualice la hora actual.

Ayuda:

- Para realizar este ejercicio y conseguir visualizar la fecha y la hora en el formato pedido, puede serte útil alguna de estas clases: Calendar, GregorianCalendar o Date.
- Para contabilizar los intervalos de 1 segundo puedes usar la clase Thread. En cualquier caso, también es posible realizar este ejercicio sin necesidad de ella.

9.13.2.- Applet Calendario de Cumpleaños

Construir un applet "Calendario" que se comporte como el que se muestra en la figura 9.11

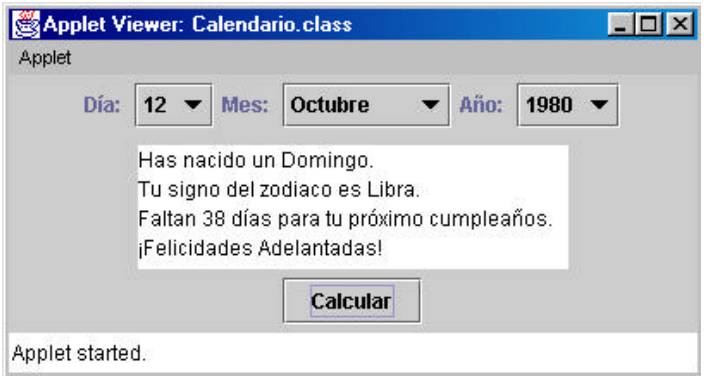

Figura 9.11: Funcionamiento del applet Calendario

El funcionamiento de este applet es el siguiente:

- El usuario introduce en las listas desplegables su fecha de nacimiento y pulsa el botón "Calcular".
- El applet calcula y visualiza:
    1. El día de la semana que se corresponde con la fecha de nacimiento.
    2. El signo del zodiaco que corresponde a esa fecha.
    3. Contando desde la fecha actual, el número de días que restan hasta el próximo cumpleaños.

Ayuda:

- La clase Calendar o GregorianCalendar puede serte muy útil para trabajar con fechas.
- Si quieres usar controles de la librería SWING (en lugar de controles AWT) es conveniente que heredes de la clase JApplet (en lugar de la clase Applet).