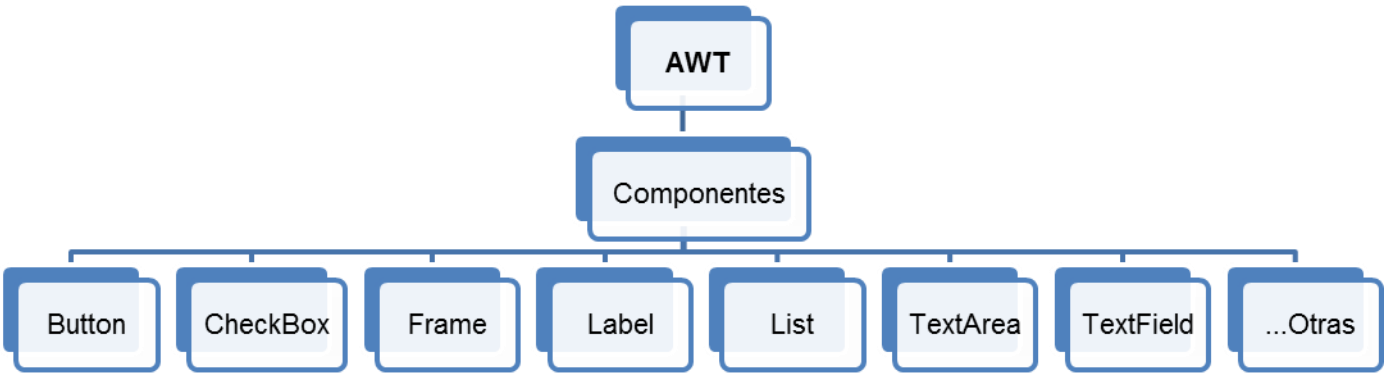
3. Manejo de eventos

Al construir una interfaz gráfica en la programación orientada a objetos, se hace imprescindible el manejo de eventos, pues por lo general se requiere incorporar acciones que sucedan en el applet o aplicación gráfica para que sea operada por el usuario. El oprimir un botón, el seleccionar un menú, el deslizar una barra de control, son ejemplos de eventos, y en este material de formación lo que se va hacer es profundizar el tema utilizando el paquete AWT de Java. A continuación se presenta su estructura:

Estructura de la clase AWT:



Tema 1. Evento:

Se refiere a cierta acción que puede ser realizada por el usuario para que el applet ejecute una determinada serie de instrucciones. Algunos eventos dentro de un applet son:

- Oprimir un botón definido.
- Mover una barra de desplazamiento.
- Apretar una tecla específica.

Cuando un applet está ejecutándose, este puede estar esperando a que el usuario active cualquier secuencia de instrucciones mediante la selección de algún elemento de interfaz gráfica definida para esto, tal y como se ha hecho anteriormente con el objeto de clase button.

Cuando se crea un objeto de interfaz gráfica, la meta es que el usuario indique alguna acción; es importante definirle a éste, el cual será escuchado por la clase específica que se encarga de revisar si hay algún evento que ha sido realizado.

Escuchadores de eventos (listeners)

Un escuchador (listener) es un objeto que es notificado cuando un evento ocurre. Este tiene dos requerimientos:

1. Debe ser registrado con una o más fuentes que reciben notificaciones sobre tipos de eventos específicos.

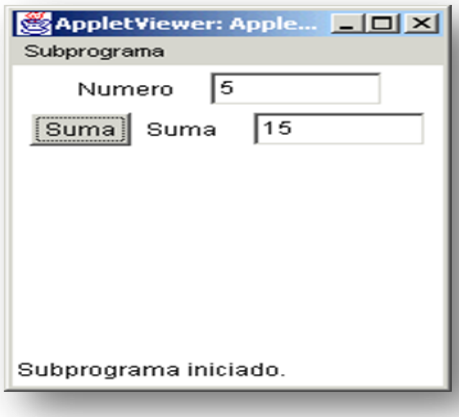
2. Necesita implementar métodos que reciban y procesen estas notificaciones.

Los métodos que reciben y procesan estas notificaciones están definidos en un grupo de interfaces encontradas en java.awt.event. Por ejemplo la clase MouseMotionListener define dos métodos que reciben notificaciones cuando el ratón es arrastrado o movido. Algún objeto puede recibir y procesar uno o ambos de estos eventos si éste provee una implementación de la interface.

Eventos de botón

A continuación se analiza lo que se hace para manejar más de un botón; se usa el applet para dar la suma de todos los números desde el 1 hasta la N:

Applet números de Fibonacci



Fuente: SENA

Suponer que ahora se quiere tener un botón que limpie los valores de los campos para volver a dar nuevos valores. El applet quedaría muy parecido al anterior hecho, sólo que ahora en el método Action Performed se debe revisar cual fue el botón que se seleccionó, y esto sería a través de la instrucción de decisión IF, y el método getSource(), tomado de la clase ActionEvent, el cual es el parámetro que pasa al método ActionPerformed. Dentro de este parámetro es donde se guarda la información sobre el elemento gráfico que fue seleccionado.

Es importante no olvidar que para poder usar las clases de eventos, es necesario llamar el paquete event, utilizando la siguiente declaración:

```
import java.awt.event.*;
```

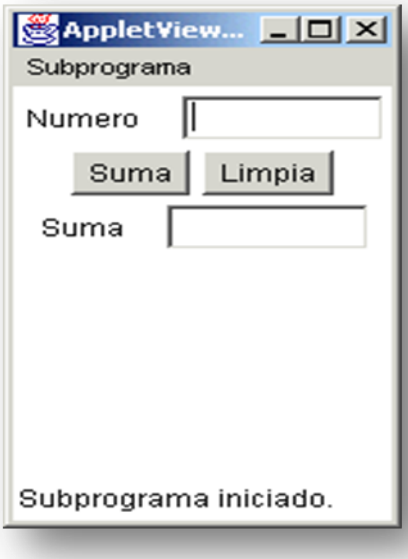
El applet quedaría como el siguiente:

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

// <applet width="150" height="200" code="AppletEventos1"></applet>
public class AppletEventos1 extends Applet implements ActionListener {
 Label l1, l2;
 TextField t1,t2;
 Button b1,b2;

 public AppletEventos1() {
 l1 = new Label("Numero");
 t1 = new TextField(8);
 l2 = new Label("Suma");
 t2 = new TextField(8);
 b1 = new Button("Suma");
 b2 = new Button("Limpia");
 add(l1);
 add(t1);
 add(b1);
 add(b2);
 add(l2);
 add(t2);
 b1. addActionListener(this);
 b2. addActionListener(this);
 }
```

```
public void actionPerformed(ActionEvent ae) {
if (ae.getSource() == b1) {
int n = Integer.parseInt(t1.getText());
int suma = 0;
for (int i = 1; i<= n; i++) {
suma += i;
}
t2.setText("" + suma);
}
if (ae.getSource() == b2) {
t1.setText("");
t2.setText("");
}
}
}
```

La ejecución del applet quedaría así:

Applet números de Fibonacci



Fuente: SENA

Lo más importante es entender el uso del método getSource() y el objeto perteneciente a la clase ActionEvent, que es el que está pasando el evento del que se trata la instrucción:

```
 if (ae.getSource() == b1) {
```

Lo que se revisa es si el evento que fue tomado corresponde al objeto del botón b1, si esto es cierto, entonces se realizan las instrucciones dentro del if, de otra manera se continúa con el siguiente if.

La misma acción se puede realizar, pero utilizando el método getActionCommand, pero aquí el cambio sería tomar el string que aparece en el elemento del botón, más no el nombre del objeto del botón. La instrucción puede ser:

```
if (ae.getActionCommand() == "Suma")
```

Las dos maneras son utilizadas para revisar cual fue el botón seleccionado, de tal manera que el applet podría quedar de la siguiente manera, funcionando igual:

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

// <applet width="150" height="200" code="AppletEventos1"></applet>
public class AppletEventos1 extends Applet implements ActionListener {
 Label l1, l2;
 TextField t1,t2;
 Button b1,b2;

 public AppletEventos1() {
 l1 = new Label("Numero");
 t1 = new TextField(8);
 l2 = new Label("Suma");
 t2 = new TextField(8);
 b1 = new Button("Suma");
 b2 = new Button("Limpia");
 add(l1);
 add(t1);
 add(b1);
 add(b2);
 add(l2);
 add(t2);
 b1. addActionListener(this);
 b2. addActionListener(this);
 }
```

```
public void actionPerformed(ActionEvent ae) {
if (ae.getActionCommand() == "Suma") {
int n = Integer.parseInt(t1.getText());
int suma = 0;
for (int i = 1; i<= n; i++) {
suma += i;
}
t2.setText("" + suma);
}
if (ae. getActionCommand() == "Limpia") {
t1.setText("");
t2.setText("");
}
}
}
```

Otro ejemplo

Ahora se muestra un ejemplo en el que se utilizan dos botones, uno para dibujar círculos en el applet al azar (empezando debajo de la coordenada 60, 80) y el otro botón es para limpiar.

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

// <applet width="150" height="200" code="AppletCirculos"></applet>
public class AppletCirculos extends Applet implements ActionListener {
 Label l1;
 TextField t1;
 Button b1,b2;
 boolean dibuja = false; // se inicializa dibuja en falso para no dibujar

 public AppletCirculos() {
 l1 = new Label("Circulos");
```

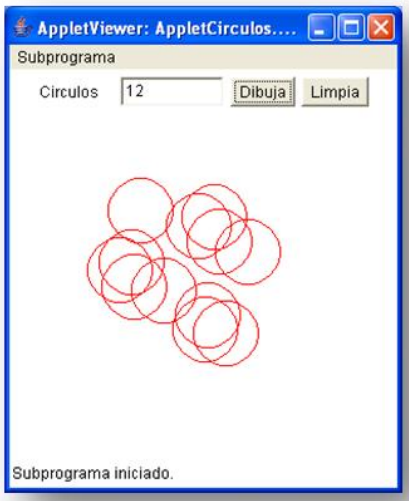
```
t1 = new TextField(8);
b1 = new Button("Dibuja");
b2 = new Button("Limpia");
add(l1);
add(t1);
add(b1);
add(b2);
b1. addActionListener(this);
b2. addActionListener(this);
}

public void actionPerformed(ActionEvent ae) {
if (ae.getSource() == b1) {
dibuja = true; // si el usuario selecciona dibujar se pone verdadero
}
if (ae.getSource() == b2) {
dibuja = false; // si el usuario selecciona limpiar se pone en falso
}
repaint();
}

public void paint(Graphics g) {
        if (dibuja) { // si dibuja es verdadero se dibuja
                g.setColor(Color.red); // se cambia a rojo el color del dibujo
                int circulos = Integer.parseInt(t1.getText()); // se toma el número del
texto
                for (int i = 1; i <= circulos ; i++) { // ciclo de círculos
                        int x1 = (int) (Math.random()*100)+60;
                        int y1 = (int) (Math.random()*100)+80;
                    g.drawOval(x1,y1, 50, 50); // se dibuja un circulo en
```

```
                                        // una posición al azar
                    }
 }
 else { // si dibuja es falso se limpia el applet
          g.setColor(Color.white);
          g.fillRect(0,0,200,200);
 }
 }
}
```

(Sánchez, 2013)

Queda así:

Applet círculos



Fuente: (Sánchez, 2013)

Referencias

- Sánchez, M. (s.f.). *Gestión de eventos*. Consultado el 24 de abril 2014, en http://odelys2003.files.wordpress.com/2013/01/1-1-javagestioneventos-100310084758-phpapp01.pdf

- Servicio Nacional de Aprendizaje, SENA. (2010). *Desarrollo de aplicaciones con interfaz gráfica, manejo de eventos, clases y objetos: Java.* Colombia: Autor.

Control del documento

	Nombre	Cargo	Dependencia	Fecha
Autor	Jorge Hernán Moreno Tenjo	Instructor	Centro Metalmecánico. Regional Distrito Capital	Mayo de 2013
Adaptación	Ana María Mora Jaramillo	Guionista - Línea de producción	Centro Agroindustrial. Regional Quindío	Mayo de 2014
	Rachman Bustillo Martínez	Guionista - Línea de Producción	Centro Agroindustrial. Regional Quindío	Mayo de 2014