Eventos

¿Qué es un evento?

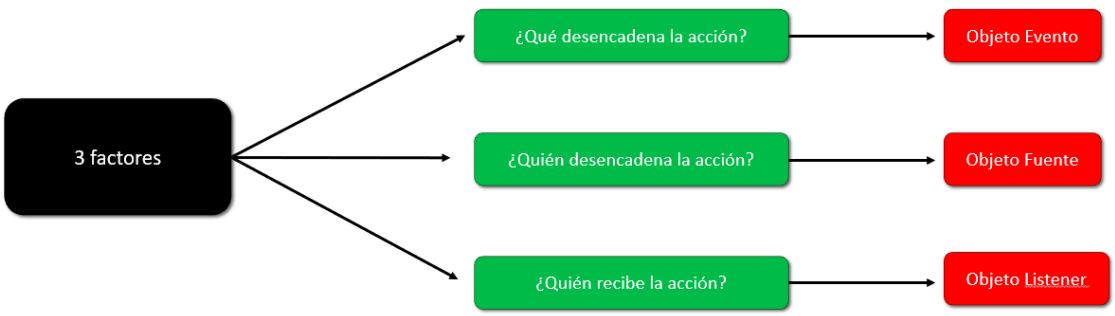
Podemos definir evento como "**desencadenante de la acción**". Cuando un usuario hace un clic en un botón y ocurre algo, o cuando por ejemplo se redimensiona o minimiza una ventana y ocurre algo, cuando se pasa el ratón sobre una imagen o un componente de otro tipo y ocurre algo… en todos estos casos se dice que ha ocurrido un **evento**. El evento es ese "algo" que desencadeno la acción: un clic en un botón, un redimensionado de ventana etc. El manejo de eventos es imprescindible para los programas que posean una interfaz gráfica.

Debido a que Java es un lenguaje orientado a objetos, la información relativa al evento está encapsulada en un objeto de tipo **evento**. Cuando ocurre un evento, se crea un objeto en tiempo de ejecución de tipo **EventObject**. En Java todos los objetos de tipo evento derivan en última instancia de la clase **java.util.EventObject**. Existen superclases para cada tipo de evento, tales como **ActionEvent** y **WindowEvent.**

Se denomina "**fuente del evento**" al objeto sobre el que se desencadenó la acción. Por ejemplo, si al hacer clic en un botón ocurre un evento (por ejemplo, que se cierre la aplicación), la fuente del evento sería el botón sobre el que se hizo clic. Si el evento ocurre al minimizar una ventana, entonces la fuente del evento sería la ventana. Las distintas fuentes de eventos pueden producir distintos tipos de eventos. Por ejemplo, un botón puede enviar objetos **ActionEvent**, mientras una ventana puede enviar **WindowEvent**.

Para que un objeto fuente del evento sea capaz de desencadenar el evento, debe estar a la "**escucha**" del evento, es decir, preparado y alerta para que en cualquier momento se realice una acción sobre él y se produzca el evento. Por ejemplo, si queremos que al pulsar un botón en concreto se cierre la aplicación, el botón debe estar a la escucha del evento clic para que cuando se pulse con el ratón sobre él, reciba (escuche) el clic y desencadene el evento.

Esquema de manejo de eventos en AWT:



Ejemplo:

JButton boton = new JButton (“OK”);

Boton.addActionListener(oyente);

A partir de este instante el objeto oyente recibe una notificación siempre que se produce un evento de acción en el botón. El evento es un clic.

El código del ejemplo anterior requiere que la clase a la que pertenece el objeto implemente la interfaz adecuada (en este caso la interfaz ActionListener).

Para implementar la interfaz ActionListener, la clase oyente debe tener un método llamado actionPerformed que recibe objetos de tipo ActionEvent como parámetro.

Ejemplo:

```
Class MiOyente implements ActionListener  {
        Public void actionPerformed (ActionEvent evento)  {
                //  reacción frente a clic del ratón
                ….
        }
}
```

Siempre que el usuario haga clic en el botón, el objeto JButton creará un objeto de tipo actionEvent y efectuará la llamada al método actionPerformed (evento), pasando al oyente este objeto de evento.

Manejo de un clic de botón

Es muy recomendable examinar con detenimiento el siguiente ejemplo ya que en él se utiliza todo lo explicado en esta lección.

Ejemplo:

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

public class PruebaBotones   {

    public static void main (String args[])  {

        MarcoBotones marco = new MarcoBotones ();

        marco.setDefaultCloseOperation  (JFrame.EXIT_ON_CLOSE); marco.setVisible(true);
    }
}


//  Marco con lámina de botones

class MarcoBotones extends JFrame    {

    public MarcoBotones ()   {

        setTitle ("Prueba botones");

        setSize (ANCHURA_PREFIJADA, ALTURA_PREFIJADA);

        // se añade la lámina al marco

        LaminaBotones lamina = new LaminaBotones();

        add(lamina);
}
    public static final int ANCHURA_PREFIJADA = 300;
    public static final int ALTURA_PREFIJADA = 200;
}

//  una lamina con tres botones

class LaminaBotones extends JPanel  {

    public LaminaBotones ()  {

        //  se crean los botones

        JButton botonAmarillo = new JButton("Amarillo");
        JButton botonAzul = new JButton("Azul");
        JButton botonRojo = new JButton("Rojo");

        //   se añaden los botones a la lamina

        add (botonAmarillo);
        add(botonAzul);
        add(botonRojo);

        //  se crean las acciones de los botones

        AccionColor accionAmarillo = new AccionColor (Color.YELLOW);
        AccionColor accionAzul = new AccionColor (Color.BLUE);
        AccionColor accionRojo = new AccionColor (Color.RED);

        //   se asocian las acciones a los botones

        botonAmarillo.addActionListener(accionAmarillo);
        botonAzul.addActionListener(accionAzul);
        botonRojo.addActionListener(accionRojo);
    }

    //  oyente de acciones que establece el color de fondo de la lamina

    private class AccionColor implements ActionListener  {

        public AccionColor (Color c)  {

            colorDeFondo = c;
        }

        public void actionPerformed (ActionEvent evento)  { setBackground(colorDeFondo);

        }

        private Color colorDeFondo;
    }
}
```