# EVENTOS EN SWING

**Evento ActionListener**

# Eventos en Swing

- Los eventos están organizados en jerarquías de clases de eventos:
  - Fuentes de Eventos (Source) Es un objeto que tiene la capacidad de detectar eventos y notificar a los receptores de eventos que se han producido estos eventos
  - Receptor de Eventos (Listener) Es un objeto que está preparado para ser notificado de la ocurrencia de un evento. Una vez que el objeto receptor está registrado para ser notificado de esos eventos, el suceso de un evento en esta clase automática, invocará al método sobrescrito del objeto receptor.

# Eventos en Swing

| Acción que produce el evento | Tipo de oyente |
|---|---|
| El usuario pulsa un botón, presiona Return mientras teclea en un campo de texto, o elige un ítem de menú. | **ActionListener** |
| El usuario elige un frame (ventana principal). | **WindowListener** |
| El usuario pulsa un botón del ratón mientras el cursor está sobre un componente. | **MouseListener** |
| El usuario mueve el cursor sobre un componente. | **MouseMotionListener** |
| El componente se hace visible. | **ComponentListener** |
| El componente obtiene obtiene el foco del teclado. | **FocusListener** |
| Cambia la tabla o la selección de una lista. | **ListSelectionListener** |

# Eventos en Swing

- Algunas clases de eventos, como los de ratón, involucran a un determinado conjunto de eventos diferentes. Una clase receptora de eventos que implemente el interfaz que recoja estos eventos debe sobrescribir todos los métodos declarados en el interfaz.
- Para prevenir esto, de forma que no sea tan tedioso y no hay que sobrescribir métodos que no se van a utilizar, se han definido un conjunto de clases intermedias, conocidas como Adaptadores (**Adapter**)

# Evento: ActionListener

- en caso de tener que detectar el botón que origina un evento entre varios botones registrados con un ActionListener, puede ser útil recurrir al método setActionCommand, que establece un nombre, independiente de su etiqueta, para el botón:

  - boton1.setActionCommand ("nombre_ActionCommand1");
  - boton2.setActionCommand ("nombre_ActionCommand2");

# Evento: ActionListener

| **Eventos, interfaces y métodos de escucha** | **Componentes que lo generan** |
|---|---|
| ActionEvent<br>ActionListener<br>addActionListener() | JButton, JList, JTextField,<br>JmenuItem,<br>JCheckBoxMenuItem, JMenu,<br>JpopupMenu |

# Evento ActionListener: Distintas opciones

- DISTINTAS VERSIONES:
  - El objeto  llama a  ActionListener
  - La clase pricipal implementa ActionListener
  - La clase principal no implementa ActionListener pero sí lo hace otra clase:

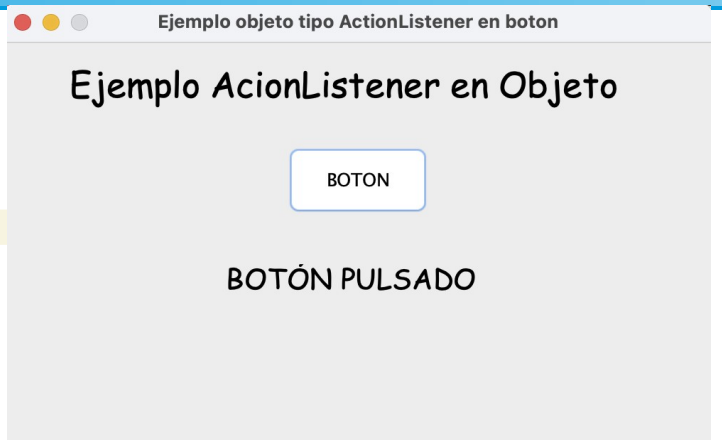
# Evento ActionListener: Distintas opciones

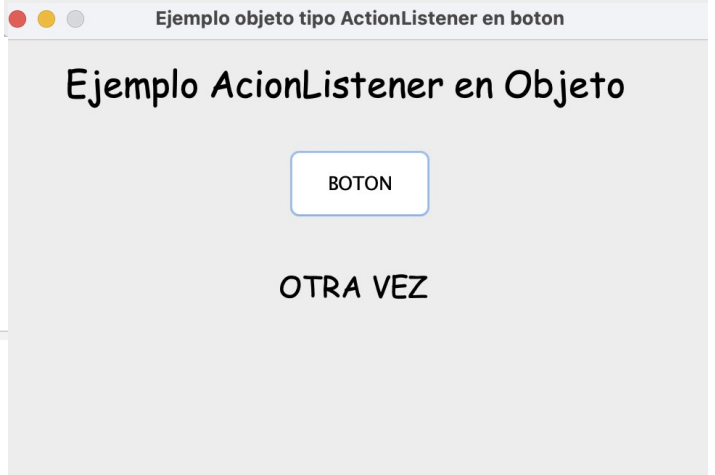
- El objeto  llama a ActionListener:

```
boton1 = new JButton();
boton1.setBounds(x: 150, y: 75, width: 100, height: 50);
boton1.setText(text:"BOTON");

//  Otra opcion:
    boton1.addActionListener(e -> {
        System.out.println(x: "hola");
    });

/*
boton1.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent e) {
        System.out.println("hola");
        if ("BOTÓN PULSADO".equals(etiqueta1.getText())) {
            etiqueta1.setText("OTRA VEZ");

        } else {
            etiqueta1.setText("BOTÓN PULSADO");
        }
    }
});
```

# Evento ActionListener: Distintas opciones

```
boton1 = new JButton();
boton1.setBounds(x: 200, y: 75, width: 100, height: 50);
boton1.setText(text: "BOTON");

ActionListener oyente = new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        System.out.println(x: "hola");
        if ("BOTÓN PULSADO".equals(anObject: etiqueta1.getText())) {
            etiqueta1.setText(text: "OTRA VEZ");

        } else {
            etiqueta1.setText(text: "BOTÓN PULSADO");
        }
    }
};
```



# Evento ActionListener: Distintas opciones

- DISTINTAS VERSIONES:
- • la clase pricipal implementa ActionListener:

```
……. boton.addActionListener(this);
……. public void actionPerformed(ActionEvent e) {
    ... código ... ;
}
```

# Evento ActionListener: Con Lambda

```
boton1 = new JButton();
boton1.setBounds(x: 150, y: 75, width: 100, height: 50);
boton1.setText(text: "BOTON");

//  Otra opcion:
    boton1.addActionListener(e -> {
        System.out.println(x: "hola");
    });

/*
boton1.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent e) {
        System.out.println("hola");
        if ("BOTÓN PULSADO".equals(etiqueta1.getText())) {
            etiqueta1.setText("OTRA VEZ");

        } else {
            etiqueta1.setText("BOTÓN PULSADO");
        }
    }
});
```

# Evento ActionListener: Distintas opciones

- En caso de tener que detectar el botón que origina un evento entre varios botones registrados con un ActionListener, puede ser útil recurrir al método **setActionCommand**, que establece un nombre, independiente de su etiqueta, para el botón:

```
boton1.setActionCommand ("nombre_ActionCommand1");
boton2.setActionCommand ("nombre_ActionCommand2");
```

```
public void actionPerformed(ActionEvent e) {
   ……..

    if ("nombre_ActionCommand1".equals(e.getActionCommand())) {
     ... código de que se apreto por ejemplo el boton 1 ... ; }
    else if
         ("nombre_ActionCommand2".equals(e.getActionCommand()))
          {
              ... código del clic del boton 2 ... ; }
          }
```

# Evento ActionListener: Distintas opciones

```
        boton1.addActionListener(new ProcesoAccion());

        add(comp: labelTitulo);
        add(comp: etiqueta1);
        add(comp: etiqueta2);
        add(comp: boton1);

        setTitle(title: "Ejemplo ActionListener");
        setSize(width: 400, height: 440);
        setResizable(resizable: false);
        setDefaultCloseOperation(operation: EXIT_ON_CLOSE);
        setLocationRelativeTo(c: null);
        setVisible(b: true);
    }

    class ProcesoAccion implements ActionListener {

        public void actionPerformed(ActionEvent e) {
            //Me muestra por consola el componente que invocó el evento
            System.out.println("evt.getActionCommand() = " + e.getActionCommand());

            if (e.toString().indexOf(str:"on CampoTexto") != -1) {
                System.out.println(x: "Capturado actionPerformed sobre el objeto CampoTexto");
            }
            if (e.toString().indexOf(str:"on Boton") != -1) {
                System.out.println(x: "Capturado actionPerformed sobre el objeto Boton");
            }
        }
    }
```