PRÁCTICA 8.- Etiqueta que funciona como liga o un enlace

Objetivo

Desarrollar un nuevo componente a partir de la clase *JLabel*, modificándolo de manera que se comporte como una liga a un programa o una página web.

Objetivos específicos:

- Se diseñará el nuevo componente usando herencia de *JLabel*.
- Se le habilitarán eventos de ratón para que cambie de color y se cambie el cursor.
- Se le habilitará el evento de Acción o de Clic del Mouse para que responda abriendo la aplicación que se tiene mencionada en la etiqueta.

Introducción

Esta práctica nos permite aplicar los conocimientos aprendidos en la generación de nuevos componentes. Es un refuerzo a este tema presentando un nuevo caso de adecuación de un componente existente al que se le modifica su comportamiento de manera que responda a nuestras necesidades y que el componente original no tiene manera de hacerlo.

Se parte del componente *JLabel* (etiquetas), el cual será adaptado para que se comporte como un texto de conexión o enlace. El componente detectará cuando el cursor pase sobre él y entonces, cambiará de color, se subrayará y el cursor cambiará a la imagen de una mano, que es la que representa que hay alguna acción a realizar en ese lugar.

En principio se hará que el componente active aplicaciones que están en la propia computadora como son el block de notas, la calculadora o el paint.

Correlación con los temas y subtemas del programa de estudio vigente

Se relaciona directamente con los temas de la Unidad II ya que se hace uso de la interface gráfica de desarrollo y se generan aplicaciones gráficas.

Con la Unidad III tiene una gran relación dado que se utilizan componentes que ya existen en el entorno de desarrollo, se crean nuevos componentes y adicionalmente, se utilizan librerías ya definidas en el ambiente.

Material y equipo necesario

- Computadora o laptop.
- Software requerido: NetBeans 7.2 en adelante.

Metodología

Se creará un nuevo componente que se comporte como una liga a un programa de manera que cambie de color, se modifique el cursor y se subraye cuando pase el cursor por encima de ella. Luego enlazará con la aplicación o programa definido cuando se detecte el *mouseClik* sobre el componente.

Paso 1. Abrir un nuevo proyecto de Java Application con el nombre de *Practica8_LigaPrograma* desmarcando la casilla de verificación del Create Main Class.

Paso 2. En la ventana proyectos, sobre *Source Packages* activar el menú contextual y añadir una nueva *Java Class* con nombre *Liga*. El código generado es el siguiente:

```
public class Liga {



}
```

Paso 3. Hacemos que la clase *Liga* herede de la clase *JLabel* y de la interface *MouseListener*. Modificar la clase a que diga lo siguiente:

```
public class Liga extends JLabel implements MouseListener{

}
```

En este punto hay dos cosas por hacer para eliminar los errores que nos marca el editor: en el menú contextual escoger *Fix Imports* y lo segundo que se nos observa es que no se han implementado TODOS los métodos de la interface *MouseListener*. Para corregirlo lleve el cursor al borde izquierdo sobre el botoncito

rojo que indica que hay error, dar clic sobre él y escoja la opción *Implement All Abstract Methods*. El código se verá así:

```
public class Liga extends JLabel implements MouseListener{
   @Override
   public void mouseClicked(MouseEvent e) {
      throw new UnsupportedOperationException("Not supported yet.");
   }

   @Override
   public void mousePressed(MouseEvent e) {
      throw new UnsupportedOperationException("Not supported yet.");
   }

   @Override
   public void mouseReleased(MouseEvent e) {
      throw new UnsupportedOperationException("Not supported yet.");
   }

   @Override
   public void mouseEntered(MouseEvent e) {
      throw new UnsupportedOperationException("Not supported yet.");
   }

   @Override
   public void mouseExited(MouseEvent e) {
      throw new UnsupportedOperationException("Not supported yet.");
   }
}
```

Enseguida elimine la instrucción “throw new….” de cada uno de los métodos aportados por la interface. Una manera rápida de hacerlo es llevar el cursor a la línea que se desea eliminar y con las teclas Ctrl + E se elimina dicha línea.

Paso 4. Añadimos un constructor a nuestra clase *Liga*. El camino ya se indicó en la práctica 7.

```
public Liga() {

  }
```

Escribir dentro del constructor la instrucción para añadir el *MouseListener*. Esto es indispensable para que se puedan detectar los eventos del mouse y responder a ellos.

```
public Liga() {

    addMouseListener(this);

  }
```

Paso 5. Declaramos dos variables de apoyo a nuestro programa. Una variable booleana de nombre *subraya* que identificará si hay que subrayar o no la etiqueta y la otra para respaldar el color de la etiqueta y poderlo restablecer cuando deje de estar activada la etiqueta,(el cursor encima de la etiqueta). El código debe verse así:

```
public class Liga extends JLabel implements MouseListener{
  private boolean subraya= false;
  private Color colorAnte;

  public Liga() {
    addMouseListener(this);
  }
  @Override.....
```

Hacer un *Fix Imports* para corregir el error en la palabra Color.

Paso 6. Abajo del constructor sobrescribimos el método *paint()*. Recuerde activar menú contextual, seleccionar *Override Method*, seleccionar el método *paint()* en la nueva ventana y crearlo.

```
@Override

  public void paint(Graphics g) {

    super.paint(g);

  }
```

Hay que modificar este método para cuando la variable *subraya* esté en verdadera, pinte una línea en la parte de abajo del componente. Para ello modifique el método paint() de la siguiente manera:

```
@Override
  public void paint(Graphics g) {
    super.paint(g);
    if(subraya)
      g.drawLine(2,getHeight()-2, getWidth()-2, getHeight()-2);
  }
```

Paso 7. Añadimos una nueva propiedad para definir el color del subrayado de la etiqueta. La forma de añadir una propiedad ya se vio en la práctica anterior, así que solo diremos que tiene por nombre: *colorSubrayado*; tipo de dato: *Color*; valor inicial: *Color.magenta*; especificador de ámbito: *private*.

Después de haber creado la nueva propiedad el código que se añadió es el siguiente:

```
    private static Color colorSubrayado = Color.magenta;
      public static Color getColorSubrayado() {
        return colorSubrayado;
      }

      public static void setColorSubrayado(Color colorSubrayado) {
        Liga.colorSubrayado = colorSubrayado;
      }
```

Paso 8. Se creará otra propiedad a nuestro componente. Nombre: *comando*; valor inicial: "*calc.exe*"; tipo de dato: *String*; especificador de ámbito: *private*.

Después de hacer lo anterior se creó la nueva propiedad:

```
    private String comando = "calc.exe";

      public String getComando() {
        return comando;
      }
      public void setComando(String comando) {
        this.comando = comando;
      }
```

Paso 9. Queremos que la etiqueta cambie de color y se subraye cuando pase el cursor encima de ella. También nos interesa que el cursor cambie de imagen de una flecha a la de una mano. El evento de mouse que detecta cuando el cursor está encima del elemento es el *mouseEntered*. Lo programamos de la siguiente manera:

```
public void mouseEntered(MouseEvent e) {
    colorAnte= getForeground();
    setForeground(getColorSubrayado());
    setCursor(new Cursor(Cursor.HAND_CURSOR));
    subraya = true;
    repaint();
  }
```

Paso 10. Al salir el cursor del área de la etiqueta, se deben de restablecer las condiciones iniciales. El evento a programar es el *mouseExited*.

```
public void mouseExited(MouseEvent e) {
    setForeground(colorAnte);
     setCursor(new Cursor(Cursor.DEFAULT_CURSOR));
     subraya = false;
    repaint();
  }
```

Paso 11. El siguiente evento del mouse que nos interesa es la acción que va a realizar al hacer clic sobre él. En este caso queremos que ejecute el programa que tenemos definido en la propiedad comando.

```
public void mouseClicked(MouseEvent e) {
     try {
       Runtime runtime = Runtime.getRuntime();
       runtime.exec(getComando());
     } catch (IOException ex) {
       Logger.getLogger(Liga.class.getName()).log(Level.SEVERE, null, ex);
     }
  }
```

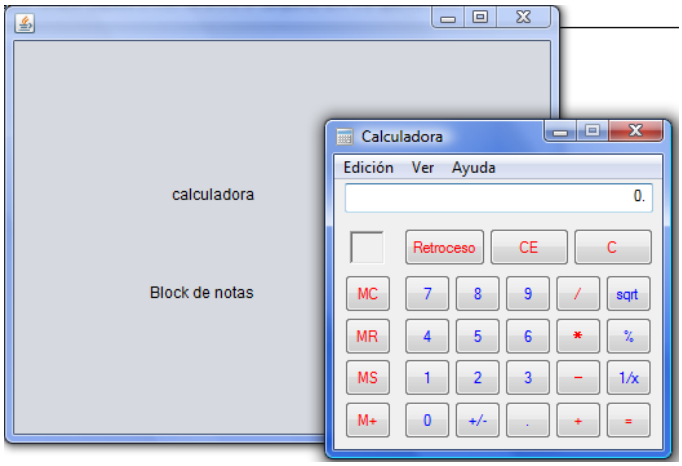
Paso 12. Se procede a compilar el componente y a incorporarlo a la paleta de componentes. Estos pasos se conocen, por lo que ya no se describen con detalle. Remítase a la práctica anterior para apoyo en caso de dudas.

Paso 13. Añadir un nuevo *JFrame Form* al proyecto para poder probar el componente liga creado. Le damos por nombre al formulario nuevo el de *VentanaLiga*.

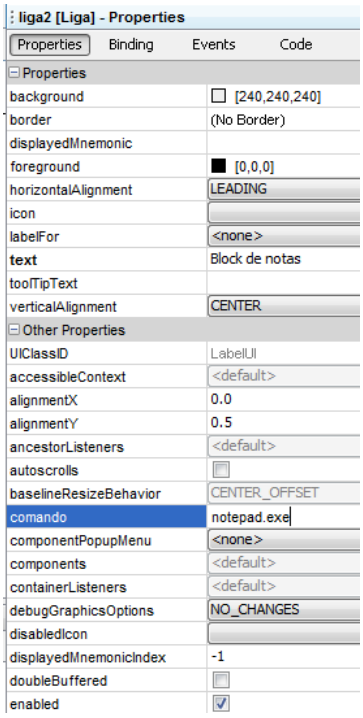
Paso 14. Ponemos cuatro componentes liga en nuestro formulario. Les cambiamos el texto a *calculadora*, *block de notas* y *paint*, a tres de ellos.

Paso 15. Ejecute el programa y cosntate que al pasar el mouse por encima de cualquiera de los componentes, se observa que efectivamente cambia de color,se subraya y también que cambia el tipo de cursor. De igual manera cuando el cursor deja de estar encima de la etiqueta, ésta regresa a su estado original y el cursor a su tipo por default.

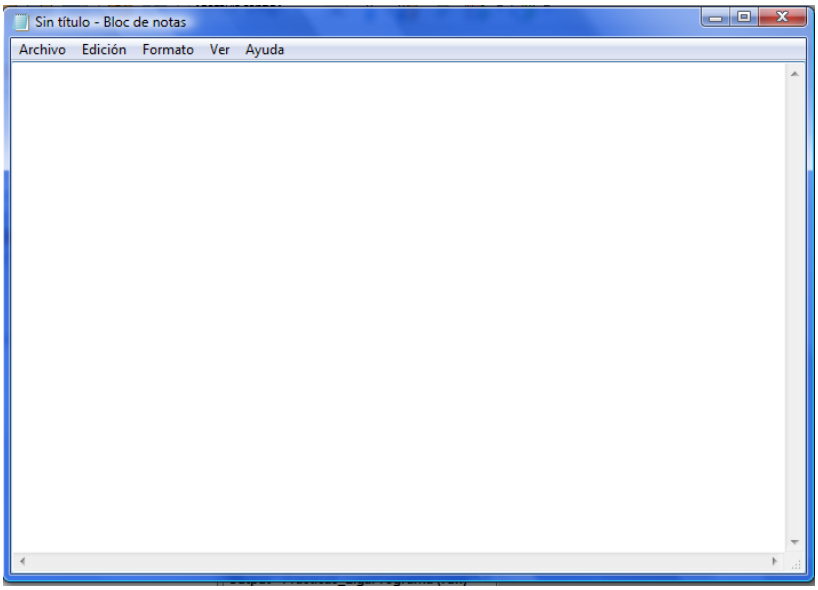
Paso 16. En este momento si se hace clic a cualquiera de los componentes, TODOS abrirán la calculadora, ya que la propiedad *comando* tiene el valor "calc.exe" por defecto.

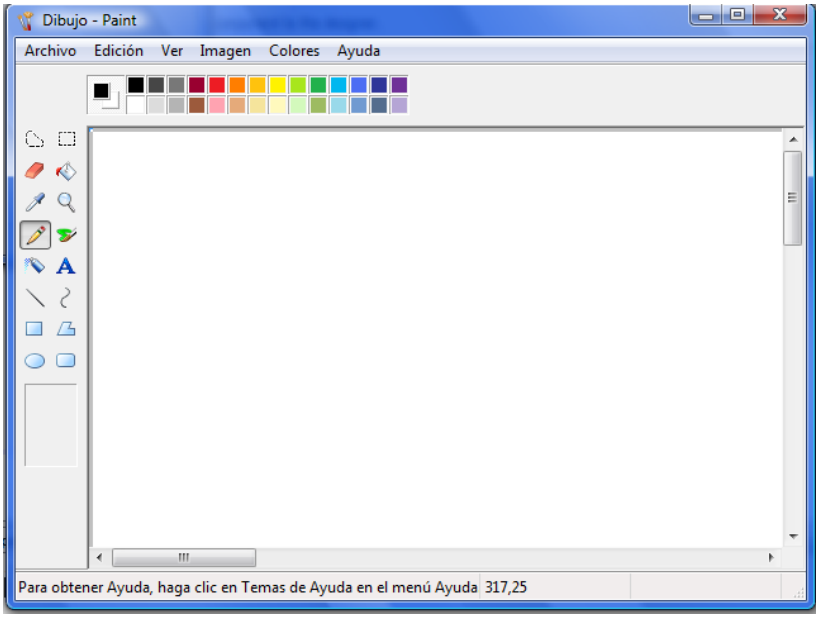


Paso 17. Al componente con nombre *block de notas* cambiele la propiedad comando por "*notepad.exe*" y al que dice paint, cámbiela por el de "*mspaint.exe*".



Ejecute el programa y observe lo que sucede al hacer clic en cada componente.





Paso 19. Al cuarto componente con texto actualmente de liga 4, le asignaremos un programa ejecutable que tengamos en otro directorio distinto al que contiene los programas *calc.exe*, *mspaint.exe* y *notepad.exe* (C: Windows/System32), así que lo buscamos y copiamos TODA su trayectoria y la pegamos en la propiedad comando de nuestra cuarta etiqueta. Pruebe a abrir un juego desde esta etiqueta. En la computadora de trabajo usada para desarrollar esta práctica se tiene el siguiente programa:

"C:\Archivos de programa\Trellix2\program\TRELLIX.EXE"

Sugerencias Didácticas

En este caso las sugerencias son las mismas de la práctica 7 debido a que es continuación del tema ahí presentado.

Hacer énfasis en la importancia de contar con la competencia de crear nuevos componentes y del mundo de posibilidades que se les abre, al no estar limitados exclusivamente a lo que ofrece el entorno de desarrollo.

Reporte del alumno (resultados)

El alumno deberá presentar evidencias suficientes de haber realizado la práctica.

Reportar sus conclusiones y recomendaciones.

Bibliografía preliminar

1. Java The Complete Reference Eighth Edition
   Herbert Schildt

   Oracle Press 2011

2. Java 2 Interfaces Gráficas y Aplicaciones para Internet
   Francisco Javier Ceballos
   Alfaomega- RAMA 2ª. Edición 2007

3. En la página oficial de Java:
   docs.oracle.com/javase/tutorial/java/