Diseño de Interfaces de Usuario

Nombre: Zabai Armas Herrera

Curso: 4º

Índice:

Práctica 2. Proyectos con Arrastre/Dragging

Controlar eventos del ratón.

Base:

1. Creación de los componentes necesarios de la interfaz: panel de capas, paneles, fondo de imagen, etiqueta de autor, etc…

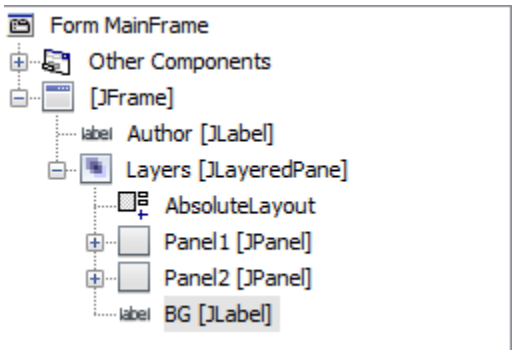


Figura 1. Elementos de la interfaz

2. Variables para controlar el espacio y las posiciones del ratón, eventos de ratón pressed y dragged con el perfeccionismo de cambiar el cursor cuando se presiona un panel.

```
292        // Variables de intercambio de informacion entre eventos
293        // Dimensiones del espacio y de la primera capa
294        private int wBox, hBox, wPanel1, hPanel1, wPanel2, hPanel2;
295
296        private int x0Panel1, y0Panel1, x0Panel2, y0Panel2, x0Mouse, y0Mouse;
```

Figura 2. Variables de control

```
138        private void Panel1MousePressed(java.awt.event.MouseEvent evt) {
139            // TODO add your handling code here:
140            // Obtiene las dimensiones actuales del contenedor y del objeto
141            // Se actualiza por si existe resize
142            Rectangle r = Layers.getBounds();
143            wBox = r.width;
144            hBox = r.height;
145            r = Panel1.getBounds();
146            wPanel1 = r.width;
147            hPanel1 = r.height;
148
149            // Obtiene la posición actual del panel
150            x0Panel1 = r.x;
151            y0Panel1 = r.y;
152
153            // Obtiene la posición actual del ratón
154            Point p = Layers.getMousePosition();
155            x0Mouse = p.x;
156            y0Mouse = p.y;
157
158            Panel1.setCursor(new java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
159        }
```

Figura 3. Evento de presión

```
161     private void Panel1MouseDragged(java.awt.event.MouseEvent evt) {
162         // TODO add your handling code here:
163         // Obtener la posición actual del ratón
164         Point p = Layers.getMousePosition();
165         if(p == null) return;
166 
167         // Calcular la nueva posición del panel
168         int xPanel1 = x0Panel1 + p.x-x0Mouse;
169         int yPanel1 = y0Panel1 + p.y-y0Mouse;
170 
171         // Evita que salga del área de trabajo
172         if(xPanel1 < 0) xPanel1 = 0;
173         if(yPanel1 < 0) yPanel1 = 0;
174 
175         int xMax = wBox - wPanel1;
176         int yMax = hBox - hPanel1;
177         if(xPanel1 > xMax) xPanel1 = xMax;
178         if(yPanel1 > yMax) yPanel1 = yMax;
179 
180         // Actualiza la posición del panel
181         p.x = xPanel1;
182         p.y = yPanel1;
183         Panel1.setLocation(p);
184     }
185 
186     private void Panel1MouseReleased(java.awt.event.MouseEvent evt) {
187         // TODO add your handling code here:
188         Panel1.setCursor(new java.awt.Cursor(java.awt.Cursor.DEFAULT_CURSOR));
189     }
```

Figura 4. Evento de arrastre y liberación

3. Añadir una imagen de fondo al panel de capas. En este caso la imagen elegida es esta:

Figura 5. Fondo

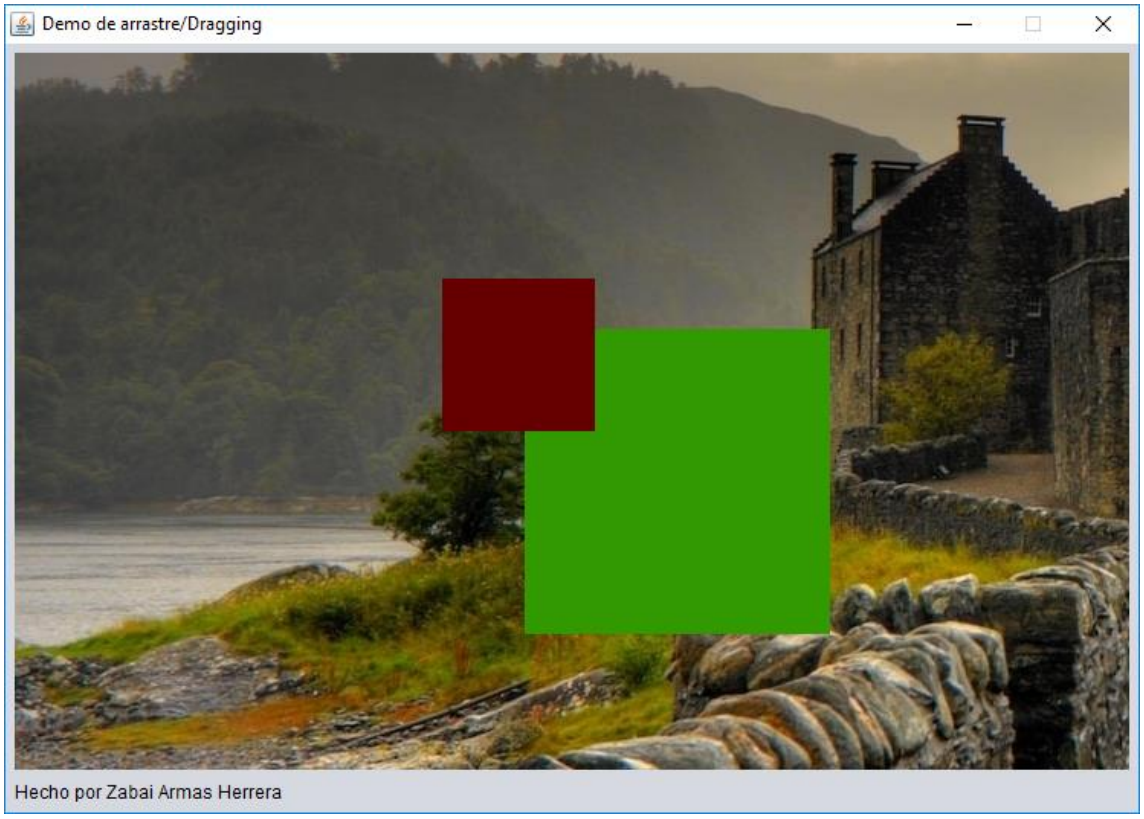
4. Crear un segundo panel reproduciendo los eventos del primero, pero con diversas funcionalidades extra. Estas por ejemplo son: cambiar su profundidad hacia arriba con el click izquierdo y con el derecho lo devuelve al fondo.

```
191     private void Panel2MouseClicked(java.awt.event.MouseEvent evt) {
192         // TODO add your handling code here:
193         int button = evt.getButton();
194         if(button == MouseEvent.BUTTON1) Layers.moveToFront(Panel2);
195
196         if(button == MouseEvent.BUTTON3) Layers.moveToFront(Panel1);
197     }
```

Figura 6. Cambio de profundidad del segundo panel

Si es necesario la comprobación de código o ejecución del mismo, este proyecto y todas las demás prácticas pueden encontrarse en el repositorio GitHub del autor.

Capturas:



Bibliografía:

- PDF de la práctica
- Documentación de Java
    - Eventos del ratón
    - Como hacer un listener del ratón
- Stack Overflow