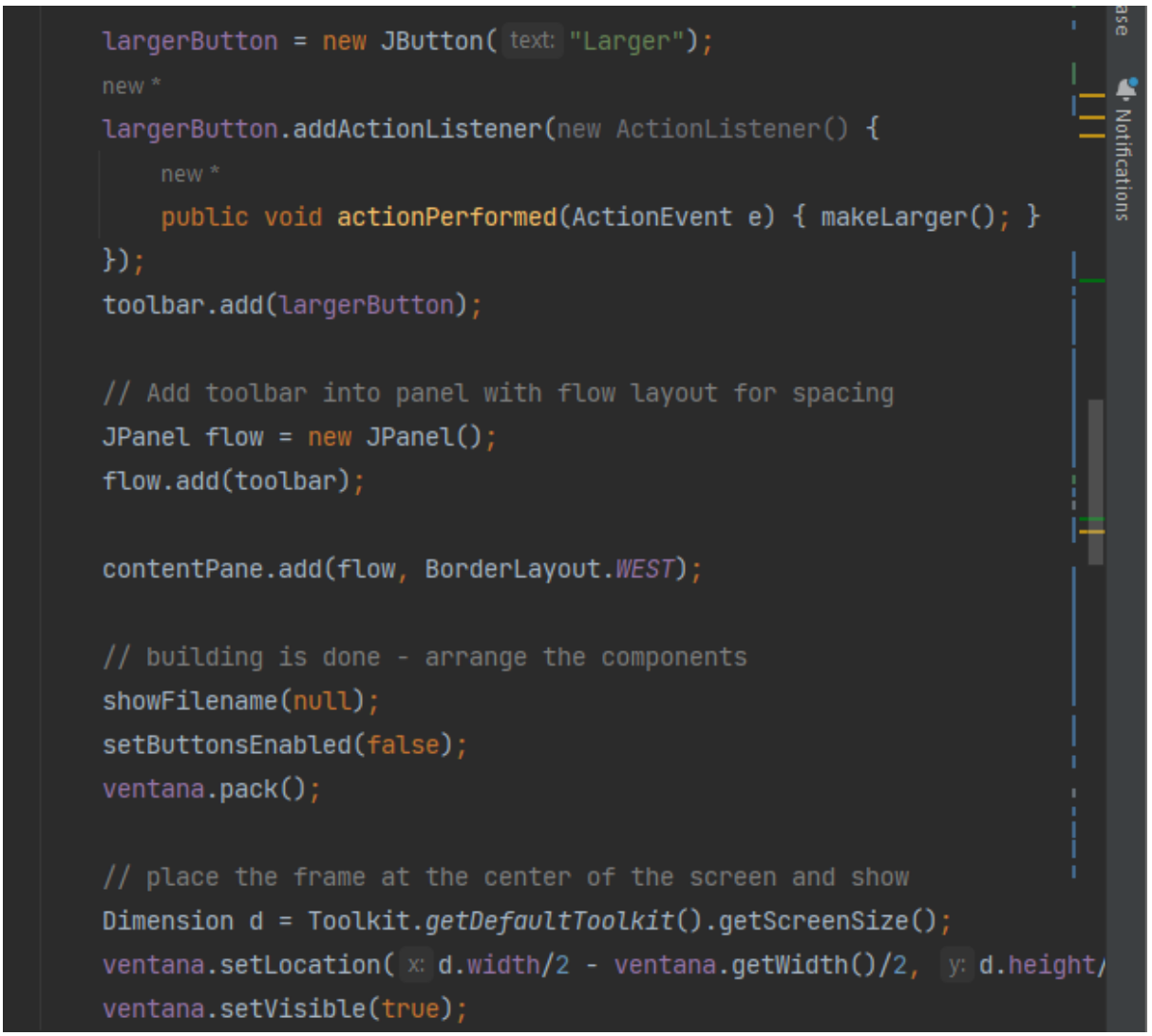
Ejercicio 43

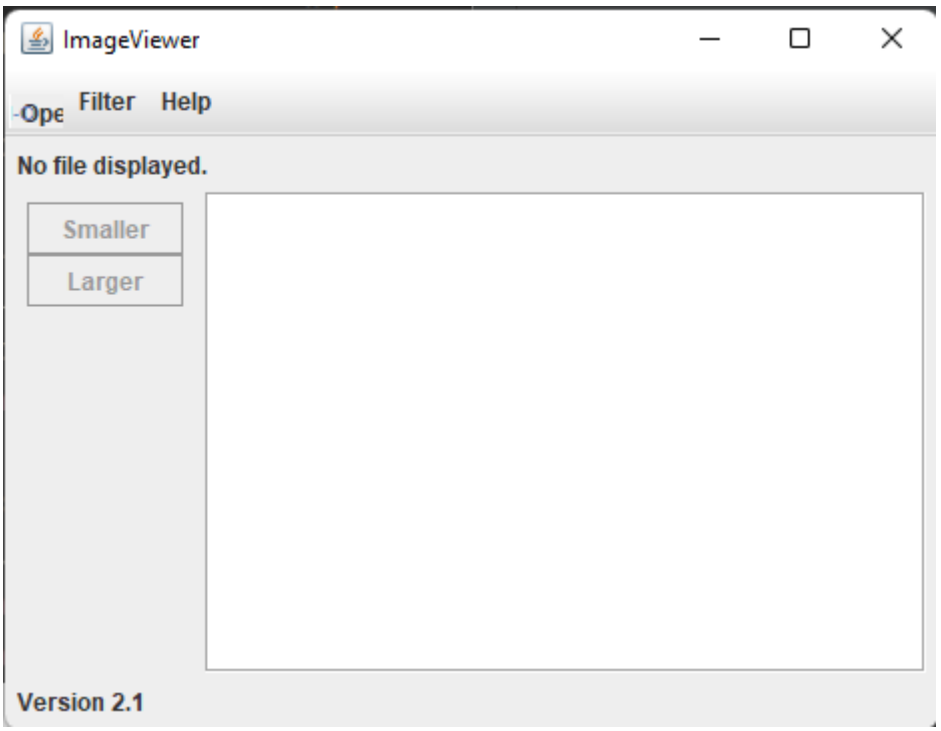
```
private void construirVentana()
{
    ventana = new JFrame( title: "ImageViewer");
    JPanel contentPane = (JPanel)ventana.getContentPane();
    contentPane.setBorder(new EmptyBorder( top: 6,  left: 6,  bottom: 6,  righ

    construirMenu(ventana);

    // Specify the layout manager with nice spacing
    contentPane.setLayout(new BorderLayout( hgap: 6,  vgap: 6));

    // Create the image pane in the center
    panelDeImagen.setBorder(new EtchedBorder());
    contentPane.add(panelDeImagen, BorderLayout.CENTER);

    // Create two labels at top and bottom for the file name and status
    etiquetaNombreDeArchivo = new JLabel();
    contentPane.add(etiquetaNombreDeArchivo, BorderLayout.NORTH);

    etiquetaEstado = new JLabel(VERSION);
    contentPane.add(etiquetaEstado, BorderLayout.SOUTH);

    // Create the toolbar with the buttons
    JPanel toolbar = new JPanel();
    toolbar.setLayout(new GridLayout( rows: 0,  cols: 1));

    smallerButton = new JButton( text: "Smaller");
    new *
    smallerButton.addActionListener(new ActionListener() {
        new *
        public void actionPerformed(ActionEvent e) { makeSmaller(); }
    });
    toolbar.add(smallerButton);

    largerButton = new JButton( text: "Larger");
    new *
```



```
largerButton = new JButton( text: "Larger");
new *
largerButton.addActionListener(new ActionListener() {
    new *
    public void actionPerformed(ActionEvent e) { makeLarger(); }
});
toolbar.add(largerButton);

// Add toolbar into panel with flow layout for spacing
JPanel flow = new JPanel();
flow.add(toolbar);

contentPane.add(flow, BorderLayout.WEST);

// building is done - arrange the components
showFilename(null);
setButtonsEnabled(false);
ventana.pack();

// place the frame at the center of the screen and show
Dimension d = Toolkit.getDefaultToolkit().getScreenSize();
ventana.setLocation( x: d.width/2 - ventana.getWidth()/2,  y: d.height/
ventana.setVisible(true);
```

Ejercicio 44

Al declarar el menu con un new Gridlayout(0,1), se desajustan los botones, este layout se crean, espacios para posibles otros items, es por ello que se desajustan los items.



Ejercicio 45, Ejercicio 46, Ejercicio 47

Agrandar:

```
private void agrandar()
{
    if(currentImage != null) {
        // create new image with double size
        int width = currentImage.getWidth();
        int height = currentImage.getHeight();
        ImagenOF newImage = new ImagenOF( width: width * 2,  height: height

        // copy pixel data into new image
        for(int y = 0; y < height; y++) {
            for(int x = 0; x < width; x++) {
                Color col = currentImage.getPixel(x, y);
                newImage.setPixel( x: x * 2,  y: y * 2, col);
                newImage.setPixel( x: x * 2 + 1,  y: y * 2, col);
                newImage.setPixel( x: x * 2,  y: y * 2 + 1, col);
                newImage.setPixel( x: x * 2+1,  y: y * 2 + 1, col);
            }
        }

        currentImage = newImage;
        panelDeImagen.setImage(currentImage);
        ventana.pack();
    }
}
```

Empequeñecer:

```java
private void empequeñecer()
{
    if(currentImage != null) {
        // create new image with double size
        int width = currentImage.getWidth() / 2;
        int height = currentImage.getHeight() / 2;
        ImagenOF newImage = new ImagenOF(width, height);

        // copy pixel data into new image
        for(int y = 0; y < height; y++) {
            for(int x = 0; x < width; x++) {
                newImage.setPixel(x, y, currentImage.getPixel( x: x * 2,
            }
        }

        currentImage = newImage;
        panelDeImagen.setImage(currentImage);
        ventana.pack();
    }
}
```

```java
2 usages  new *
private void setButtonsEnabled(boolean status)
{
    smallerButton.setEnabled(status);
    largerButton.setEnabled(status);
}
```

```java
setButtonsEnabled(true);
```

Ejercicio 49

Other UI-Related Trails

Although this is the main trail for learning about GUIs, it isn't the only trail with UI-related information.

- 2D Graphics, which describes the 2D graphics features available in the JDK.
- Sound, which discusses the sound capabilities available in the JDK.
- Java Applets, which describes API available only to applets.
- Essential Java Classes, which covers many topics, including properties and the standard I/O streams.
- The JavaFX Documentation, which describes how to build UIs with JavaFX.
- The Bonus trail contains Full-Screen Exclusive Mode API, a lesson that describes how to use API introduced in v1.4 to render graphics directly to the screen.

Ejercicio 50

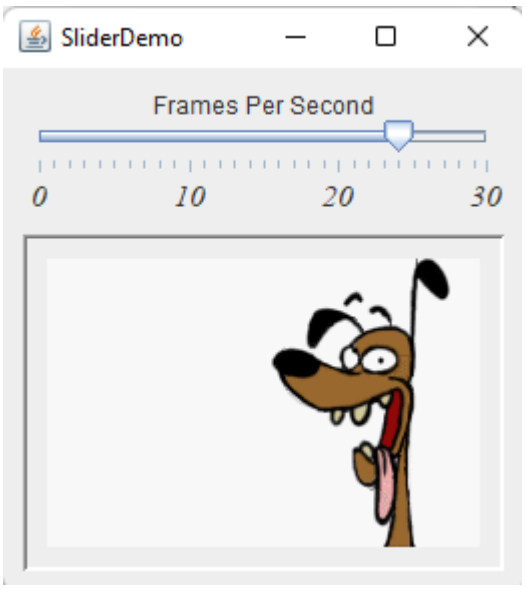
Lista de los gestores de disposicion o managers de disposición:

- FlowLayout
- BorderLayout
- CardLayout
- GridLayout
- GridBagLayout
- GroupLayout
- SprintLayout
- BoxLayout
- OverlayLayout
- CustomLayout

Ejercicio 51

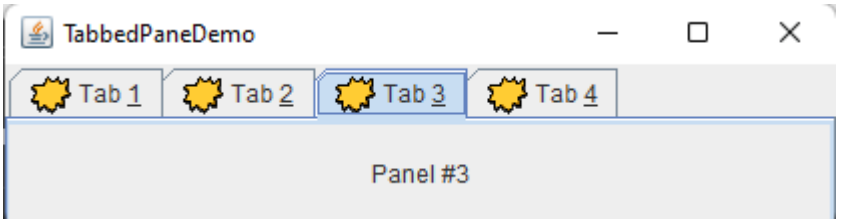
Un slider es un widget utilizado para poder obtener un valor de entre 0 y 1, este suele ser utilizado para modificar el volumen o el brillo de un ordenador. Este funciona deslizando un icono a lo largo de una barra.

```
public SliderDemo() {
    setLayout(new BoxLayout(this, BoxLayout.PAGE_AXIS));

    delay = 1000 / FPS_INIT;

    //Create the label.
    JLabel sliderLabel = new JLabel("Frames Per Second", JLabel.CENTER);
    sliderLabel.setAlignmentX(Component.CENTER_ALIGNMENT);

    //Create the slider.
    JSlider framesPerSecond = new JSlider(JSlider.HORIZONTAL,
                                          FPS_MIN, FPS_MAX, FPS_INIT);


    framesPerSecond.addChangeListener(this);

    //Turn on labels at major tick marks.

    framesPerSecond.setMajorTickSpacing(10);
    framesPerSecond.setMinorTickSpacing(1);
    framesPerSecond.setPaintTicks(true);
    framesPerSecond.setPaintLabels(true);
    framesPerSecond.setBorder(
            BorderFactory.createEmptyBorder(0,0,10,0));
    Font font = new Font("Serif", Font.ITALIC, 15);
    framesPerSecond.setFont(font);

    //Create the label that displays the animation.
    picture = new JLabel();
    picture.setHorizontalAlignment(JLabel.CENTER);
    picture.setAlignmentX(Component.CENTER_ALIGNMENT);
    picture.setBorder(BorderFactory.createCompoundBorder(
            BorderFactory.createLoweredBevelBorder(),
            BorderFactory.createEmptyBorder(10,10,10,10)));
    updatePicture(0); //display first frame

    //Put everything together.
    add(sliderLabel);
    add(framesPerSecond);
    add(picture);
    setBorder(BorderFactory.createEmptyBorder(10,10,10,10));

    //Set up a timer that calls this object's action handler.
    timer = new Timer(delay, this);
    timer.setInitialDelay(delay * 7); //We pause animation twice per cycle
                                      //by restarting the timer
    timer.setCoalesce(true);
}
```



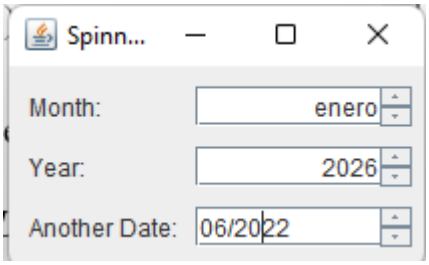
Ejercicio 52

Un jtabbedpane es un widget utilizado comúnmente por la mayoría de aplicaciones, ya que nos permiten poder separar varias secciones del programa. Este widget se basa en tener un menú en el que se hagan display diferentes botones para cambiar de sección. Puede ser usado por ejemplo para hacer display de las pestañas de un explorador web.



Ejercicio 53

Es un widget en el que podremos modificar un valor mediante unos botones en los que podremos setear el valor a antojo. Este puede de tener de diferentes tipos, un desplegable en el cual tengamos que seleccionar entre una lista o el del ejemplo de la imagen en el que deberemos de mover el valor de un valor predeterminado a otro.



Ejercicio 54

Al ejecutarse, se mostrará una barra de progreso y un botón, al presionarlo empezará a añadirse valores en la lista de progreso, esta lista tiene un textarea, en el que se van actualizando con el porcentaje actual. Este es comúnmente utilizado a la hora de instalar programas o actualizarlos.