UT5. Edición de gráficos y herramientas de dibujo: BufferedImage y Graphics. Parte 1. Imágenes

José Jarones

Clase Image

Image:

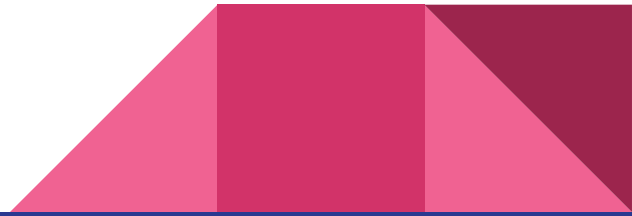
La clase abstracta Image es la superclase de todas las clases que representan imágenes gráficas. La imagen se tiene que obtener de una manera dependiente de la plataforma que estemos utilizando.

No tiene funcionalidades para la manipulación de una imagen, no guarda información sobre su representación. Sus métodos más útiles son:

getWidth()

getHeight()

getScaledInstance(int width, int height, int hints)

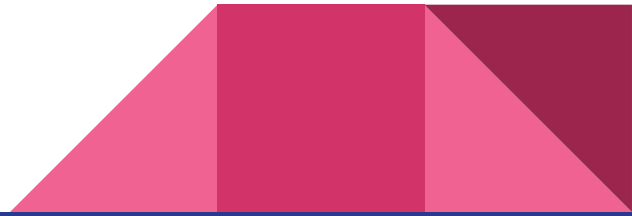


Clase BufferedImage

Es subclase de Image, describe una imagen con un buffer de imagen accesible. Está formada por un ColorModel y un Raster.

Raster: Una clase que representa un array rectangular de píxeles.

ColorModel: Es una clase que permite transcribir la información de un píxel a su valor en colores primarios y puede que transparente también. Normalmente un píxel se almacena como un int de 32 bits.



Clase ImageIcon

```
ImageIcon(Image image)
```
Creates an ImageIcon from an image object.

Con BufferImage, tenemos una clase que nos permite representar y cargar imágenes. Pero todavía no sabemos cómo pintarla en nuestros componentes de Java Swing. Para ello lo que se suele utilizar es la clase ImageIcon.

ImageIcon carga un BufferImage como un icono. Este icono puede ser incluido en un JLabel por lo que al final la imagen se mostraría por pantalla como un componente más (encapsulado en un Jlabel).

Resumen

1º Se carga la imagen

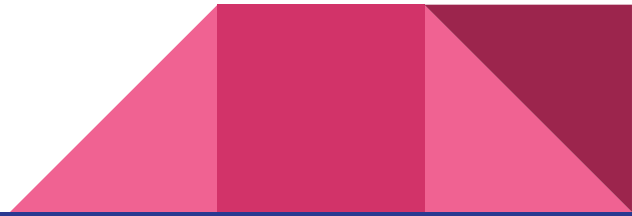
Image / BufferImage

2º Se convierte a Icono

ImageIcon

3º El icono se pmuestra en un JLabel

JLabel



Carger Imágenes

Para cargar una imagen desde un archivo específico de nuestro ordenador se utiliza el siguiente código:

```
BufferedImage img = null;
try {
        img = ImageIO.read(new File(”ruta.jpg"));
} catch (IOException e) { }
```

Uso de Imagenes

1º Se carga la imagen

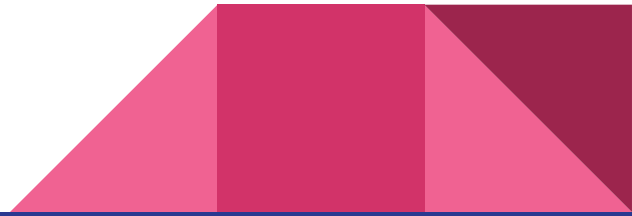
BufferedImage img = ImageIO.read(new File(path))

2º Se convierte a icono

ImageIcon icon = new ImageIcon(img);

3º El icono se muestra en JLabel

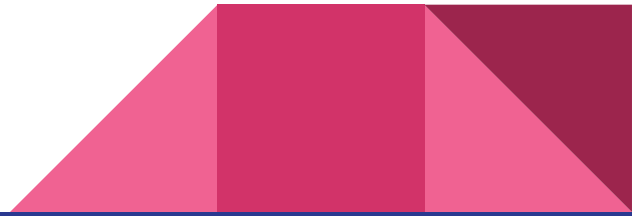
JLabel label = new JLabel(icon);



Práctica de clase 1

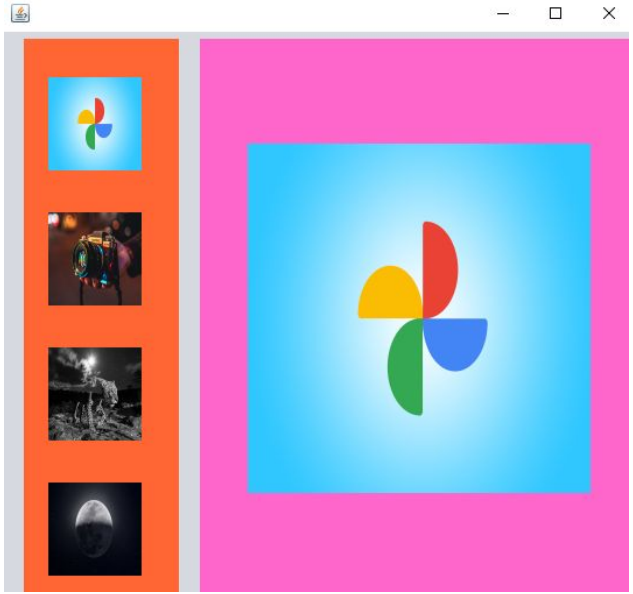
Crea un JFrame que contenga 2 imágenes.

Haz que tengan unas dimensiones de 800 x 400



Práctica de clase 2

Al pinchar una de las 4 fotos, se coloque un JLabel en el medio del panel azul con la foto pinchada.



Para cambiar el icono de un JLabel se utiliza el método:

setIcon(Icon)

Reescalar

```
public ventana() {
    initComponents();

    try{
        //Primer paso: Cargar fichero de imagen en BufferedImage
        BufferedImage img = ImageIO.read(new File("src/img/imagen1.jpg"));
        BufferedImage img2 = ImageIO.read(new File("src/img/imagen2.jpg"));
        //Segundo paso : Crear el ImageIcon
        ImageIcon icon = new ImageIcon(img);
        //Reescalar
        ImageIcon icon1 = new ImageIcon(img.getScaledInstance(300, 200, Image.SCALE_SMOOTH));
        ImageIcon icon2 = new ImageIcon(img2.getScaledInstance(300, 200, Image.SCALE_SMOOTH));
        Dimension dim = new Dimension(300,200);
        jLabel1.setSize(dim);
        jLabel2.setSize(dim);
         //Tercer paso : Poner el ImageIcon al JLabel
        this.jLabel1.setIcon(icon1);
        this.jLabel2.setIcon(icon2);
    }catch(Exception e){

        e.printStackTrace();
    }


}
```

```
public class ventana extends javax.swing.JFrame {
BufferedImage img,img2,img3,img4;
    /**
public ventana() {
    initComponents();

    try{
        //Primer paso: Cargar fichero de imagen en BufferedImage
         img = ImageIO.read(new File("src/img/imagen1.jpg"));
         img2 = ImageIO.read(new File("src/img/imagen2.jpg"));
         img3 = ImageIO.read(new File("src/img/imagen3.jpg"));
         img4 = ImageIO.read(new File("src/img/imagen4.jpg"));
        //Segundo paso : Crear el ImageIcon
      //  ImageIcon icon = new ImageIcon(img);
        //Reescalar
        ImageIcon icon1 = new ImageIcon(img.getScaledInstance(80, 80, Image.SCALE_SMOOTH));
        ImageIcon icon2 = new ImageIcon(img2.getScaledInstance(80, 80, Image.SCALE_SMOOTH));
        ImageIcon icon3 = new ImageIcon(img3.getScaledInstance(80, 80, Image.SCALE_SMOOTH));
        ImageIcon icon4 = new ImageIcon(img4.getScaledInstance(80, 80, Image.SCALE_SMOOTH));
        ImageIcon icon5 = new ImageIcon(img.getScaledInstance(300, 300, Image.SCALE_SMOOTH));

         //Tercer paso : Poner el ImageIcon al JLabel
        this.jLabel1.setIcon(icon1);
        this.jLabel2.setIcon(icon5);
        this.jLabel3.setIcon(icon2);
        this.jLabel4.setIcon(icon3);
        this.jLabel5.setIcon(icon4);
    }catch(Exception e){

        e.printStackTrace();
    }
```

Cambiar icon al pulsar clic (Botón derecho sobre JLabel ->Event->Mouse->Click)

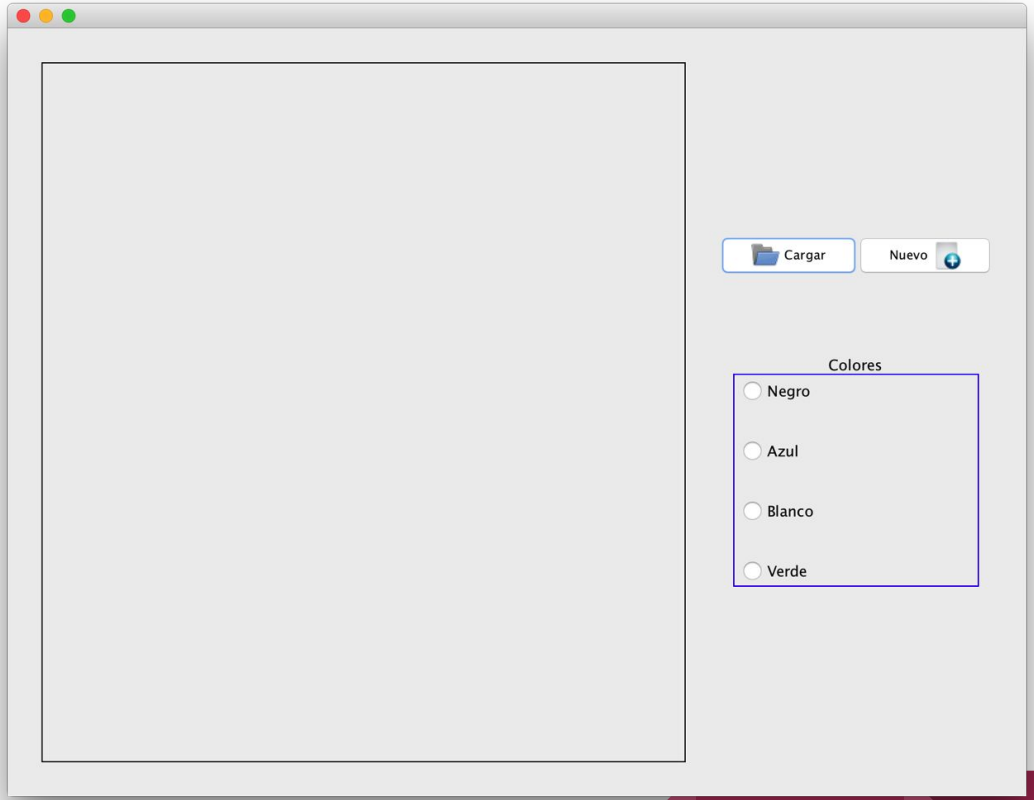
```
private void jLabel1MouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    this.jLabel2.setIcon(new ImageIcon(img.getScaledInstance(300, 300, Image.SCALE_SMOOTH)));

}
```

Práctica de clase 3

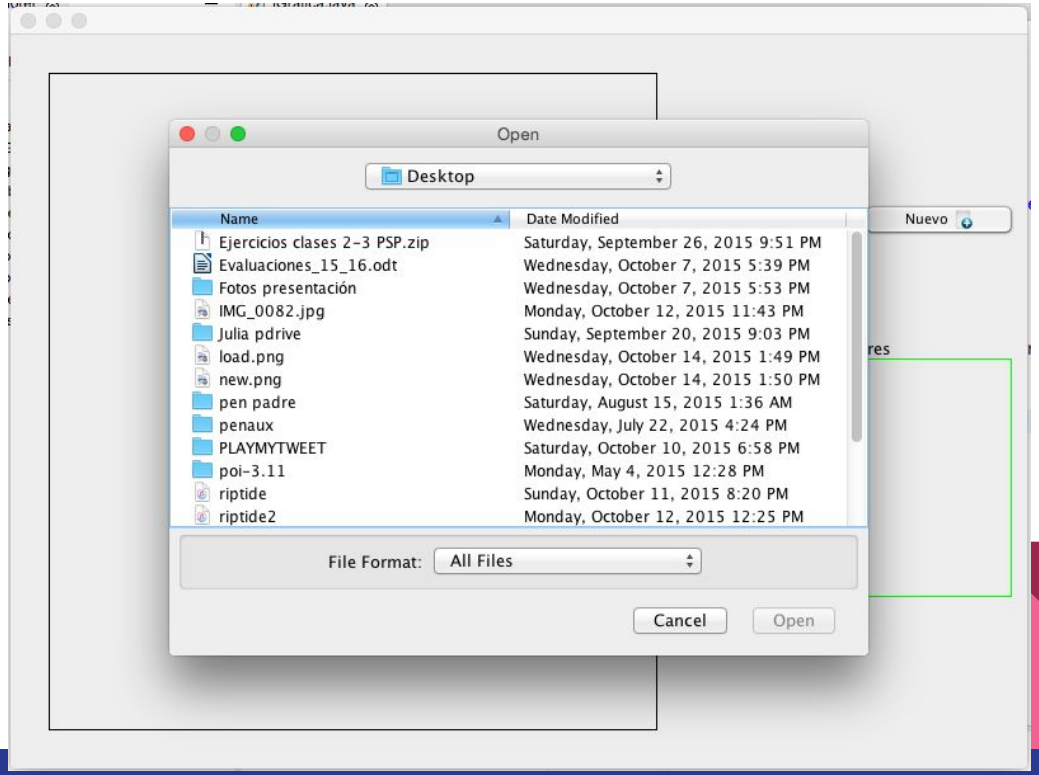
Genera el siguiente JFrame:

1. Configura el botón cargar para que cargue una imagen ocupando todo el espacio disponible en ese momento.



Práctica de clase 3

2. Al pulsar cargar que se abra un JFileChooser



Práctica de clase 3

Cuando se pulse en nuevo se coloree la etiqueta que muestra imágenes del color seleccionado por el RadioButton de la derecha

Para construir un BufferedImage desde cero utiliza el constructor:

BufferedImage(int width, int height, int imageType)

imageType normalmente es:

BufferedImage.TYPE_INT_ARGB. Para imágenes con transparentes.

BufferedImage.TYPE_INT_RGB. Para imágenes sin transparentes.

Para modificar un BufferedImage: Se utiliza la clase Graphics.

Para obtener los gráficos de una Imagen (ó un BufferedImage):

Graphics graficos = --------.getGraphics(); (-------- es la variable BufferedImage).

Para rellenar el Buffer de algún color lo que podemos hacer es pintar un rectángulo que ocupe todo el espacio. Para ello usamos el **método de Graphics**:

fillRect(int x, int y, int width, int height)

Por ejemplo:

graficos.fillRect(0, 0, 200, 200);

Pintaría un rectángulo de 200x200 en la pos 0,0

graficos.dispose() - Una vez hemos utilizado la variable la desechamos

¿Cómo pintar un BufferImage de negro de 300x300 y asignarlo a un JLabel?

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    BufferedImage buffNuevo= new BufferedImage(300,300,BufferedImage.TYPE_INT_ARGB);
    Graphics graficos = buffNuevo.getGraphics();
    graficos.setColor(Color.red);
    graficos.fillRect(0, 0, buffNuevo.getWidth(), buffNuevo.getHeight());
    graficos.dispose();
    jLabel2.setIcon(new ImageIcon(buffNuevo));
```