JOptionPane

Podemos realizar una sustitución de println por cajas de texto con JOptionPane, es una manera cómoda de usar caja de textos. Necesitamos la librería javax.swing.

Tenemos estos cuatros métodos:

Nombre del método	Descripción
showConfirmDialog	Hace una pregunta de confirmación, como sí / no / cancelar.
showInputDialog	Pedir una entrada.
showMessageDialog	Dile al usuario sobre algo que ha sucedido.
showOptionDialog	La Gran Unificación de los tres anteriores.

Todos estos métodos admiten cuatro parámetros, por orden son:

1. Componente padre. Estamos escribiendo null.
2. Mensaje de la ventana.
3. Mensaje de la barra de título.
4. Es un valor entero que hace referencia a los tipos de mensaje. Los tienes más abajo escritos. Que debe ir precedido de JOptionPane.




QUESTION INFORMATION ERROR WARNING sin icono PLAIN

Si no escribimos el cuarto parámetro sale por defecto la interrogación.

Tipo de mensaje

Define el estilo del mensaje. El administrador de Look and Feel puede diseñar el diálogo de forma diferente según este valor y, a menudo, proporcionará un icono predeterminado. Los valores posibles son:

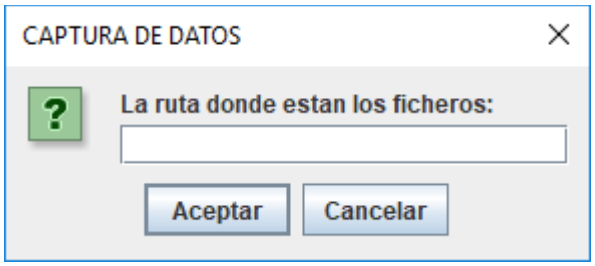
- ERROR_MESSAGE
- INFORMATION_MESSAGE
- WARNING_MESSAGE
- QUESTION_MESSAGE
- PLAIN_MESSAGE

5. Para cambiar el icono que aparece en la caja de texto, personalizando.

No tenemos que poner los cinco, solo es obligatorio el primero. Si ponemos tres parámetros en lugar de poner un título a la caja introduce ese texto en la caja de datos. Pondremos dos o cuatro, por norma.

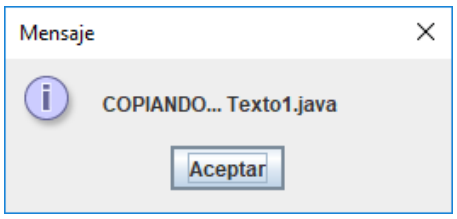
Al ejecutar la siguiente sentencia tendremos el siguiente cuadro de texto. En función del tipo de mensaje escrito nos aparecerá un icono u otro.

```
rutaorigen=JOptionPane.showInputDialog(null,"La ruta donde están los ficheros: ","CAPTURA DE DATOS",JOptionPane.QUESTION_MESSAGE);
```



OPtionPane.showMessageDialog(null, texto) Son ventanas emergentes que pueden mostrar un resultado. Hemos escrito, siendo fichero un String.

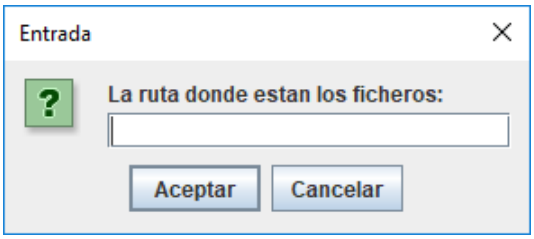
JOptionPane.showMessageDialog(null,fichero);



showInputDialog(null, texto [+ texto1 +..]), son ventanas para capturar valores. Por defecto toma los valores de tipo String si queremos capturar el valor de otro tipo tenemos que realizar la conversión correspondiente para ello. El texto escrito es el mensaje que aparece en la caja de texto.

Hemos escrito lo siguiente:

String rutaorigen=JOptionPane.showInputDialog(null,"La ruta donde estan los ficheros: ");



String cadena = (OPtionPane.showInputDialog(null, texto));

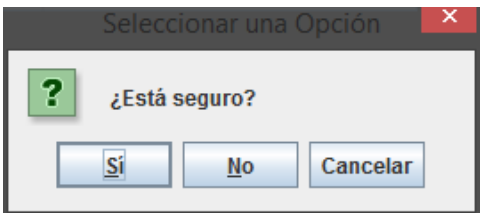
int entero = Integer.parseInt(OPtionPane.showInputDialog(null, texto));

showConfirmDialog(null, pregunta), esta opción es para confirmar el valor y tenemos la siguiente caja en ventana. Aquí podemos utilizar como cuarto parámetro el tipo de opción y como quinto parámetro el tipo de mensaje. Nos devuelve un valor numérico de tipo entero en función de la opción elegida, la primera opción empieza por el número 0 y así sucesivamente.

Tipo de opción

Define el conjunto de botones de opción que aparecen en la parte inferior del cuadro de diálogo:

- DEFAULT_OPTION
- YES_NO_OPTION
- YES_NO_CANCEL_OPTION
- OK_CANCEL_OPTION



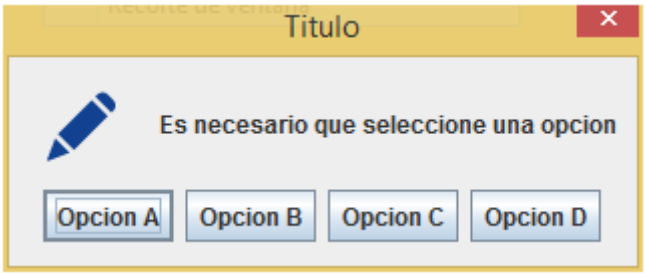
showOptionDialog, sirve para personalizar las opciones que nos salen y tiene más parámetros el método.

String [] options = {"OPCION A", "OPCION B", "OPCION C", "OPCION D"};

MyIcon icono = new MyIcon(); // para cambiar el icono esta al final de este documento la clase para usar un nuevo icono. Puedes verlo en la segunda URL.

JOptionPane.showOptionDialog(null, "ES NECESARIO QUE SELECCIONE UNA OPCION", "Titulo", JOptionPane.DEFAULT_OPTION, JOptionPane.QUESTION_MESSAGE, icono, options, options[0]);

Hemos cambiado el icono y por defecto sale marcada la primera opción de la lista.



Ejemplos:

Muestra un diálogo de error que muestra el mensaje, 'alerta':

JOptionPane.showMessageDialog(null, "alert", "alert", JOptionPane.ERROR_MESSAGE);

Muestra un diálogo de información interna con el mensaje, 'información':

JOptionPane.showInternalMessageDialog(frame, "information", "information", JOptionPane.INFORMATION_MESSAGE);

Muestre un panel de información con las opciones sí / no y el mensaje 'elija uno':

JOptionPane.showConfirmDialog(null,"choose one", "choose one", JOptionPane.YES_NO_OPTION);

Muestre un cuadro de diálogo de información interna con las opciones sí / no / cancelar y el mensaje "elija una" y la información del título:

JOptionPane.showInternalConfirmDialog(frame,"please choose one", "information",JOptionPane.YES_NO_CANCEL_OPTION, JOptionPane.INFORMATION_MESSAGE);

Obtenido de:

https://docs.oracle.com/javase/7/docs/api/javax/swing/JOptionPane.html

https://serprogramador.es/programando-mensajes-de-dialogo-en-java-parte-1/

http://chuwiki.chuidiang.org/index.php?title=JOptionPane_y_di%C3%A1logos_modales

https://www.mkyong.com/swing/java-swing-how-to-make-a-confirmation-dialog/

Definición del icono, de un tamaño de 50 x 50:

```
public class MyIcon implements Icon
{
  public void paintIcon(Component c, Graphics g, int x, int y)
  {
     Image imagen = new ImageIcon(getClass().getResource("nombre fichero.png")).getImage();
//Admite poner la ruta por delante del nombre del fichero.
     g.drawImage(imagen, x, y,c);
  }
  public int getIconWidth()   {    return 50;        }
  public int getIconHeight()   {    return 50;        }

}
```

Podemos cambiar el icono solo con esta otra opción, mucho más simple.

```
ImageIcon icon = new ImageIcon("src/images/turtle64.png");
```

Ejemplo para copiar ficheros con hilos pero utilizando cajas de texto para la captura de datos y la visualización. Pedimos ruta donde se encuentran los ficheros, ruta donde vamos a copiar y la extensión de los ficheros que deseamos copiar.
Como trabajamos con hilo hemos declarado la clase hilo para ello.

```
import java.io.*;
import javax.swing.*;

public class CopiaFicherosCaja
{
 public static void main(String [] args) throws IOException
 {
  InputStreamReader buf = new InputStreamReader(System.in);
        BufferedReader dato = new BufferedReader(buf);
        String rutaorigen, rutadestino, extension, copia;
        String [] ficheros;
        File origen, destino;
        int i;
        CrearHilo hebra;
 do{
         rutaorigen=JOptionPane.showInputDialog(null,"La ruta donde estan los ficheros: ");
         origen = new File(rutaorigen);
 }while(!origen.exists());
 do{
         rutadestino=JOptionPane.showInputDialog(null,"La ruta donde va la copia de los
ficheros: ");
         destino = new File(rutadestino);
 }while(!destino.exists());
        extension=JOptionPane.showInputDialog(null,"Indica la extension de los ficheros a
copiar ");
        ficheros = origen.list();
        for(i=0;i<ficheros.length;i++)
        {
   copia=rutaorigen;
         if(ficheros[i].endsWith(extension))
         {
     rutaorigen = rutaorigen.concat("\\"+ficheros[i]);
                 //System.out.println(ficheros[i]+".."+rutaorigen);
                 hebra = new CrearHilo(rutaorigen,rutadestino);
     hebra.start();
     rutaorigen=copia;
         }
        }
 }
}
```

```
import java.io.*;
import javax.swing.*;

class CrearHilo extends Thread
{
        String nombrecopia, dondecopiar, fichero;
        File nuevo;
        Runtime xp = Runtime.getRuntime();
        Process proceso;
        String comando = "cmd /c copy ";
        int i;
        public CrearHilo(String ori, String des)
        {
          nombrecopia = ori;
          dondecopiar = des;
          comando = comando + ori + " " + des;
        }
        public void run()
        {
                nuevo = new File(nombrecopia);

                  try{
                    proceso = xp.exec(comando);
                    for(i=nombrecopia.length()-1;i>0;i--)
                            if(nombrecopia.charAt(i)=='\\')
                                    break;
      fichero =  "COPIANDO..."+ nombrecopia.substring(i+1);
                    JOptionPane.showMessageDialog(null,fichero);
                    // sale una caja por cada uno de ellos
                  }
     catch (IOException x)
                  {System.out.println("ERROR "+nombrecopia);}
        }
}
```

Ejercicio para crear un menú de opciones para ejecutar programas desde este entorno gráfico.

```
import java.io.*;
import javax.swing.*;
import java.awt.*;
import java.util.*;
public class MenuAplicaciones
{
 public static void main(String [] args) throws IOException
 {
        Runtime pro = Runtime.getRuntime();
        String comando ="cmd /c ", resultado="";
  String [] menu ={"PAINT","CALCULADORA","BLOC DE NOTAS","ESCRITORIO
REMOTO","FIREFOX","CONSOLA DEL DOS","SALIR"};
        String [] programa ={ "mspaint","calc","notepad","mstsc","start firefox","start cmd"};
  Process proceso = null;
        ImageIcon icono = new ImageIcon("windows.jpg");

        String[]
fontNames=GraphicsEnvironment.getLocalGraphicsEnvironment().getAvailableFontFamilyNam
es();
  System.out.println(Arrays.toString(fontNames));
        int opcion;
        do{
          opcion = JOptionPane.showOptionDialog(null, "ELIGE UNA DE ELLAS",
"APLICACIONES WINDOWS",JOptionPane.DEFAULT_OPTION,
JOptionPane.QUESTION_MESSAGE,icono,menu,menu[0]);
    if(opcion<6)
          {
           resultado = comando + programa[opcion];
           proceso = pro.exec(resultado);
          }
         }while(opcion!=6);
 }
}
```