

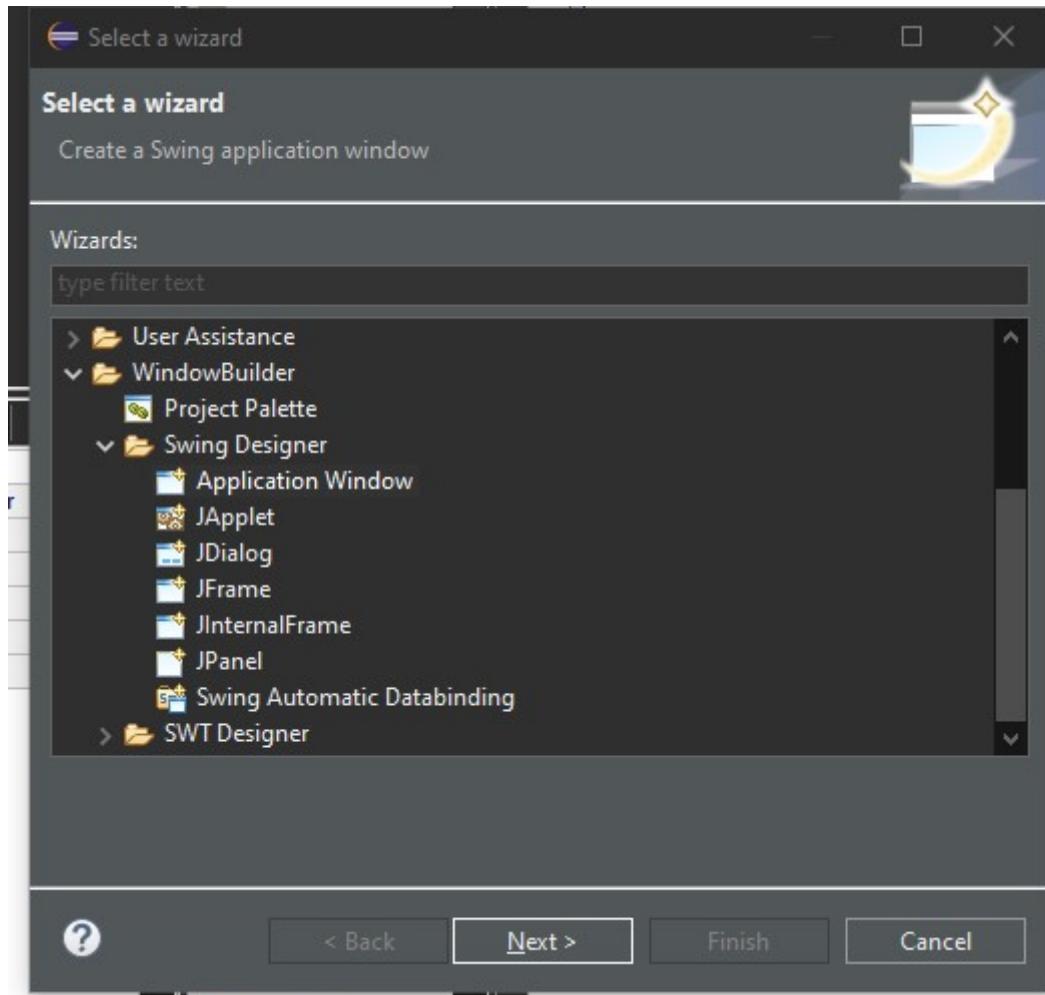
JTable

Vamos a implementar una JTable que muestre los hilos lanzados con un contador de iteraciones por Hilo y otro contador de iteraciones totales:

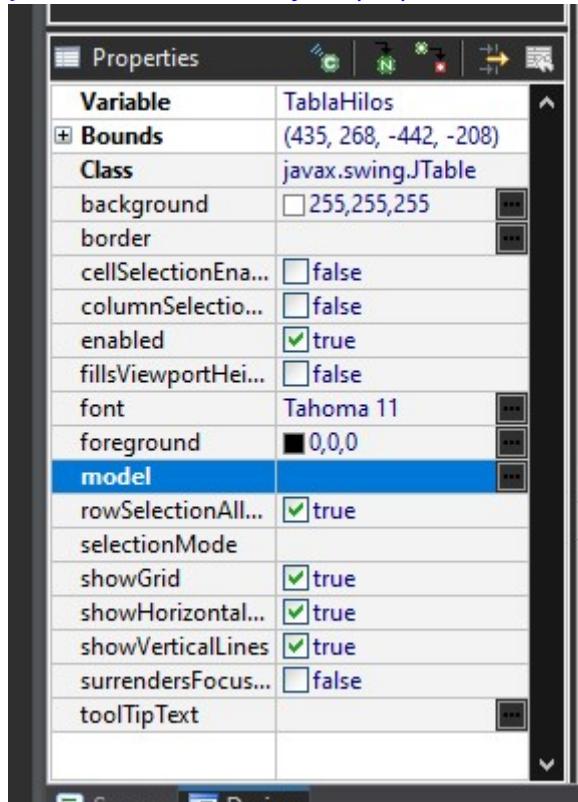


Implementación de la GUI

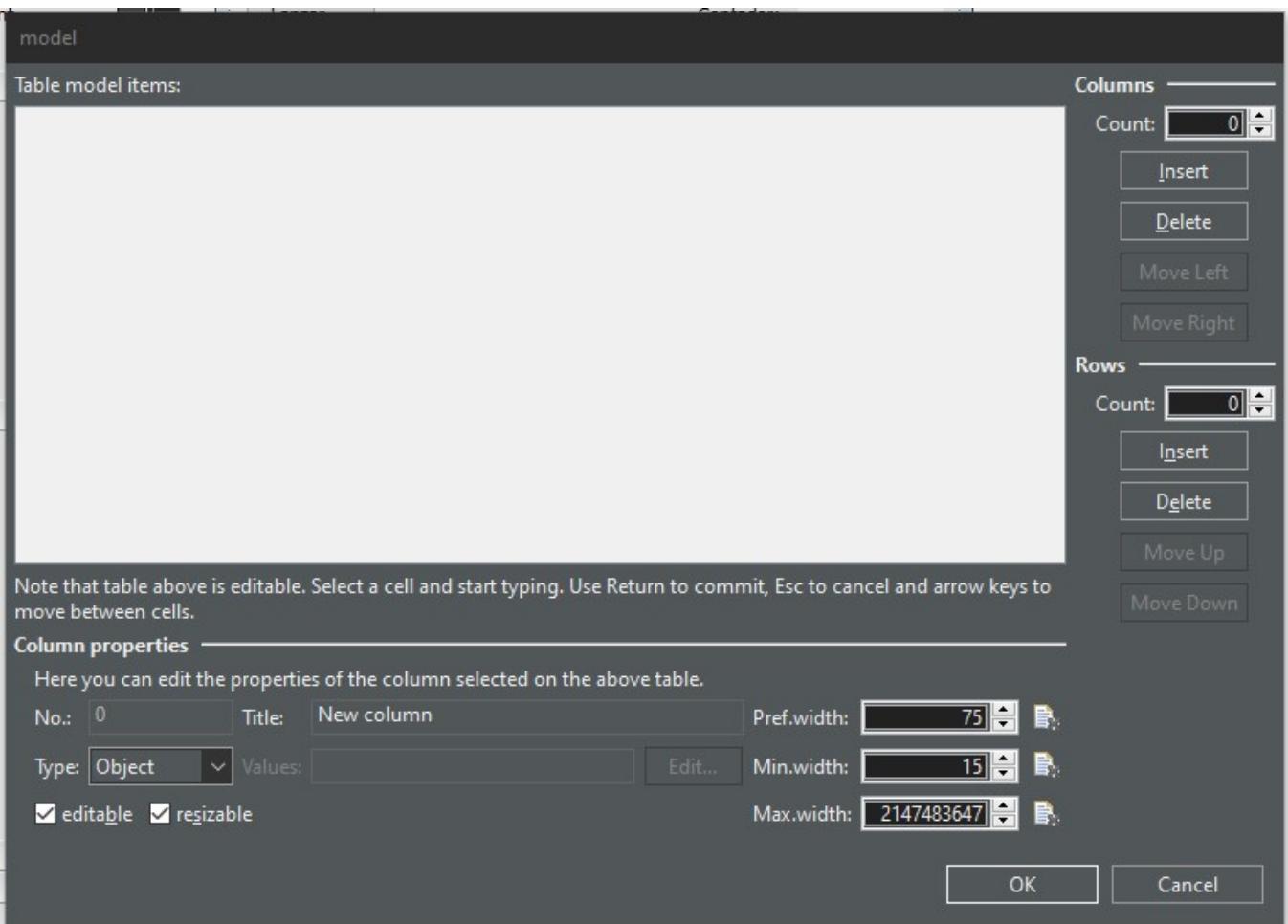
Primero creamos una nueva ventana desde el Window Builder:



A continuación insertamos la JTable en la ventana y en propiedades de esta vamos a model:



Cuando le demos a model nos saldrá la siguiente ventana con el editor de la tabla:



En la derecha le daremos a insertar una vez, colocaremos abajo el nombre de la columna, en este caso conteo, le daremos a insertar de nuevo y pondremos como nombre Hilos. Podemos ajustar el tamaño de las columnas abajo, en este caso la columna Hilos será de tamaño 200:

model

Table model items:

Hilos	Conteo
-------	--------

Columns

Count: 2

Rows

Count: 0

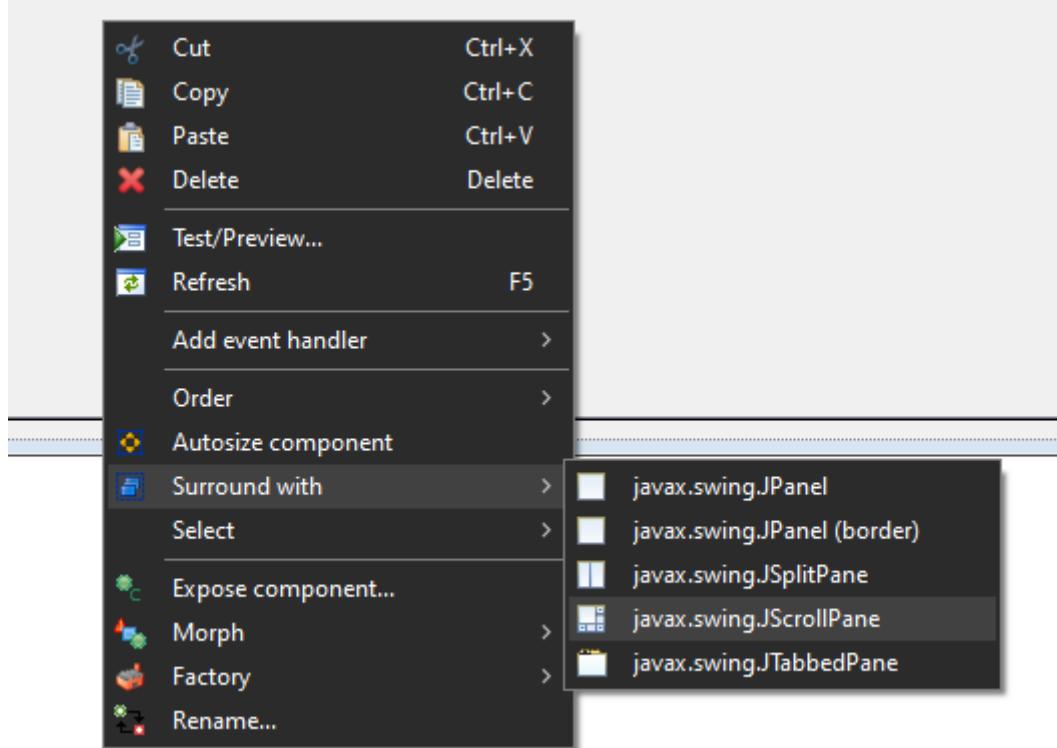
Note that table above is editable. Select a cell and start typing. Use Return to commit, Esc to cancel and arrow keys to move between cells.

Column properties

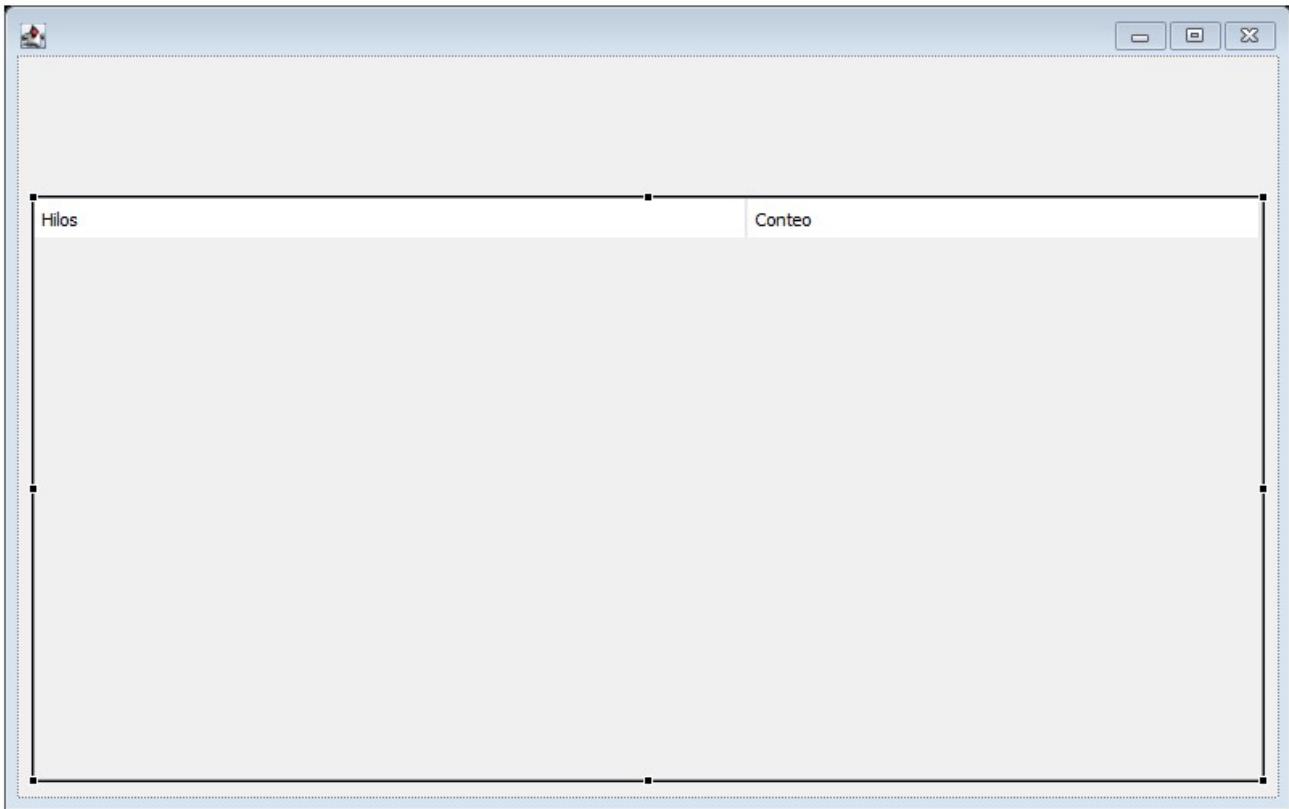
Here you can edit the properties of the column selected on the above table.

No.: 0	Title: Hilos	Pref.width: 200
Type: Object	Values:	Min.width: 15
<input checked="" type="checkbox"/> editable <input checked="" type="checkbox"/> resizable		Max.width: 2147483647

Tras darle a aceptar, no veremos nada en donde hemos insertado la tabla, ahora le daremos click derecho al recuadro donde debería estar la tabla e iremos a Surround With y seleccionaremos JScrollPane:



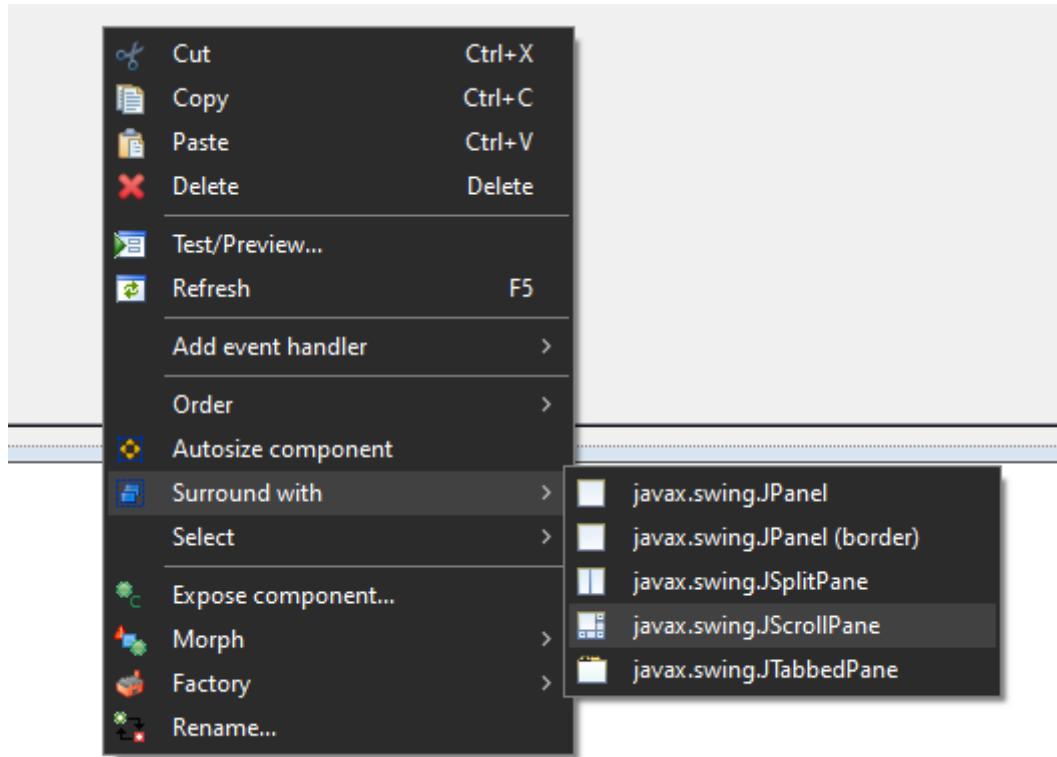
Tras eso tendremos algo como esto:



Ahora que hemos terminado con la inserción de la JTable insertaremos el Botón Lanzar y la JLabel que irá al lado del contador de iteraciones totales:

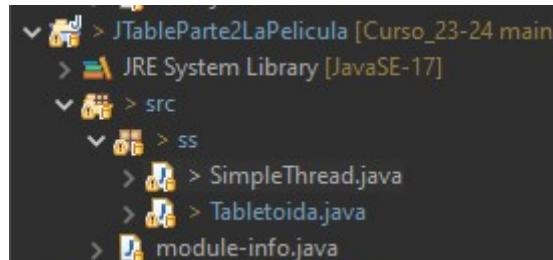


Finalmente insertaremos un JTextField no editable para que muestre los conteos totales:



Implementación Funcional

Estructura Clases



Variables SimpleThread

```
public class SimpleThread extends Thread{//Extendemos la clase Thread
    public static final int FOR_EVER = -1;//Constante para ejecución sin fin
    protected long delay;//variable para el retraso de ejecución
    protected int times;//Variable para el número de repeticiones
    protected int lecontadorInterno;//Variable para contar las iteraciones de este hilo
```

Variables Tabletoida

Hilos

Para implementar los hilos que lanzaremos crearemos una nueva clase a la que llamaremos SimpleThread:

```

1 package ss;
2
3 public class SimpleThread extends Thread{//Extendemos la clase Thread
4     public static final int FOR_EVER = -1;//Constante para ejecución sin fin
5     protected long delay;//variable para el retraso de ejecución
6     protected int times;//Variable para el número de repeticiones
7     //Constructor del hilo, indicamos el retraso que tendrá en re-ejecutarse
8     //y cuantas veces se volverá a ejecutar
9     public SimpleThread (long delay, int times) {
10         System.out.println("Constructor SimpleThread");//Texto a mostrar en consola cuando se construye un hilo
11         this.delay = delay;
12         this.times = times;
13     }
14     public void run() {
15         try {
16             for(int aux = times; (times>=0)|| (aux == FOR_EVER); times--) {
17                 System.out.println("Mi Delay es:"+delay);//Mostramos en consola el delay
18                 sleep(delay);//Para la ejecución por el tiempo de retardo especificado
19             }
20         }catch(Exception e) {
21             System.out.println("Error.");
22         }
23     }
24
25
26 }
27

```

Tras esto podemos ir añadiendo la funcionalidad al botón Lanzar para que pueda lanzar Hilos:

```

JButton botonLanzar = new JButton("Lanzar Hilo");
botonLanzar.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) { //-----Botón Lanzar (EN CONSTRUCCIÓN)
        SimpleThread hilillo = new SimpleThread(5000, 5); //nuevo objeto tipo hilo
        hilillo.start(); //Lanzamos nuevo hilo
        System.out.println("Hilo" + hilillo + "lanzado");
        contTotal++;
        contadorTotalGui.setText(""+ contTotal + ""); //cambiamos valor mostrado en JTextField, esto necesita ser modificado
    }
});
botonLanzar.setBounds(35, 11, 131, 56);
getContentPane().add(botonLanzar);

```

Sobre Actualizar la Tabla Automáticamente

Para actualizar la tabla automáticamente deberíamos pasar por referencia la tabla a la clase SimpleThread y utilizar el método: miTableModel.fireTableDataChanged().

From:

<http://knoppia.net/> - Knoppia

Permanent link:

<http://knoppia.net/doku.php?id=dad:jtable&rev=1695895769>

Last update: **2023/09/28 10:09**

