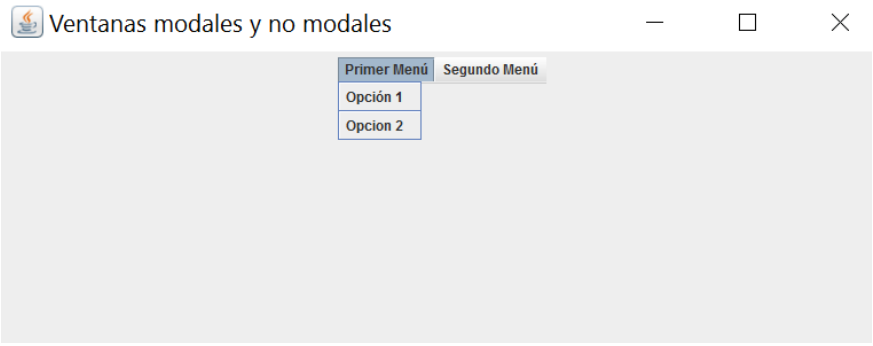
Menús.

Para crear un menú tenemos tres clases swing: JMenuBar, JMenu y JMenuItem. Con la primera creamos la barra de menús, a esta barra de menús le añadiremos las diversas opciones con JMenu y a cada JMenu le añadiremos sus items con JMenuItem. Lo mejor es construir un menú para entender el funcionamiento:

```
      //OBJETOS MENÚ:
JMenuBar menuBar;
JMenu menu;
JMenuItem menuItem;
//Instanciamos la barra de menús.
menuBar = new JMenuBar();
//La primera opción del menú.
menu = new JMenu("Primer Menú");
menuBar.add(menu);
//a la primera opción del menú le ponemos dos item.
menuItem = new JMenuItem("Opción 1");
menu.add(menuItem);
menu.addSeparator(); //separador
menuItem = new JMenuItem("Opcion 2");
menu.add(menuItem);
//Segunda opción de la barra de menús sin ningún item.
menu = new JMenu("Segundo Menú");
menuBar.add(menu);
```

Esto quedaría:



Diálogos personalizados: JDialog.

En java podemos crear ventanas que serán llamadas desde un JFrame, para ello usamos las clase JDialog. En un JDialog podemos incluir los mismos elementos que un JFrame, pero la principal diferencia es que a un JDialog hay que decirle, cuando se instancia, quién es su padre. La idea es tener una ventana principal en nuestra aplicación, un JFrame, que hará de padre del resto de ventanas necesarias que serán JDialog.

Vamos a crear un JDialog, para ser llamada desde un JFrame:

```
public class DialogoModal extends JDialog {
    private JTextField textField;

    public DialogoModal(JFrame padre) {
            //el constructor recibe como parámetro aquella ventana
            // que le ha llamado, o sea, su padre


        super(padre, true); //invoco el constructor de la case, con dos
                            //parámetros: quién es su padre
                            //después true para modal y false para no modal

        setTitle("Mete un dato");
        textField = new JTextField(20);
        add(textField);

        // Se oculta la ventana al pulsar <enter> sobre el textfield
        textField.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent arg0) {
                setVisible(false);
            }
        });
    }


    public String getText() {
        return textField.getText();
    }
}
```

Vamos a llamar esta ventana desde el JFrame que contiene el menú creado en el punto 2:

```
public class menu extends JFrame{
    public menu(){
        setSize(800,600);
        setTitle("Ventanas modales y no modales");
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLayout(new FlowLayout());
        //MENÚ:
        JMenuBar menuBar;
        JMenu menu;
        JMenuItem menuItem;
        //Create the menu bar.
        menuBar = new JMenuBar();
        //Build the first menu.
        menu = new JMenu("Primer Menú");
        menuBar.add(menu);
        //a group of JMenuItems
        menuItem = new JMenuItem("Opción 1");
        //Añadir listener a una opción del menú:
        menuItem.addActionListener(new em(this) );
        menu.add(menuItem);
        menu.addSeparator();
        menuItem = new JMenuItem("Opcion 2");
        menu.add(menuItem);
        //Build second menu in the menu bar.
        menu = new JMenu("Segundo Menú");
        menuBar.add(menu);
```

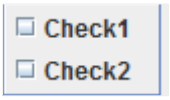
```
        add(menuBar);


    }
    //LISTENER FUERA:
    class em implements ActionListener {
            JFrame miFrame;
            public em(JFrame f){
                miFrame=f;
            }
            public void actionPerformed(ActionEvent e) {
                DialogoModal dialogoModal =new DialogoModal(miFrame);
                dialogoModal.pack();
                  // para darle tamaño automático a la ventana.
                dialogoModal.setVisible(true);
                //Como es una ventana modal no volverá a esta línea
                //hasta que no sea cerrada la otra
                System.out.println(dialogoModal.getText());
            }
    }

}
```

El código en rojo sirve para remarcar cómo es llamado el JDialog y cómo se le pasa quién es la ventana padre.

JcheckBoxMenuItem.



Este compontes es otro item que almacena el JMenu, permite vincular casillas de verificación o Checkbox, muy útil cuando vamos a parametrizar mas de una opción o característica de nuestro sistema...

```
1jCheckMenu1 = new JCheckBoxMenuItem("Check1");
2menuOpciones.add(jCheckMenu1);
```

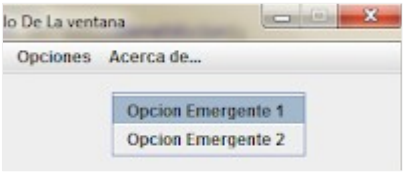
JradioButtonMenuItem.

Este componente es similar al anterior, la diferencia es que permite vincular componentes RadioButton los cuales brindan opciones de selección única, por ejemplo si en un editor de texto queremos un único tipo de letra, podemos usar este componente.

...

```
1jRadioButtonMenu1 = new JRadioButtonMenuItem("Radio1");
2grupoRadios.add(jRadioButtonMenu1);
3menuOpciones.add(jRadioButtonMenu1);
```

JpopupMenu.



Por ultimo tenemos el JPopupMenu, a diferencia de los anteriores, este componente no es contenido en la Barra de Menú, sino que se asocia al contenedor principal que para nuestro ejemplo es un JPanel (del cual hablaremos en el siguiente articulo), permite brindar opciones emergentes o popup con tan solo dar click derecho sobre algún área del panel..... el JPopup funciona también como un contenedor similar al Jmenu....

```
menuEmergente = new JPopupMenu();
itemEmergente1.setText("Opcion Emergente 1");
menuEmergente.add(itemEmergente1)
```

Ejercicio:

Crea una aplicación llamada “casa” que disponga de un menú con la siguiente distribución:

•Casa (tendrá un icono con una casa)
•Habitaciones (Menu)
•Habitación 1 (MenuItem).
•Habitación 2 (MenuItem): desactivado.
•Salón (CheckBox): seleccionado. Ctrl+S
•SEPARADOR
•Cocina (RadioButton): seleccionado. Ctrl+C
•Baño (RadioButton): Ctrl+B
•Extras
•Garaje (MenuItem): Alt+G
•Trastero (MenuItem): Alt+T
A continuación se pueden ver dos prototipos del menú: