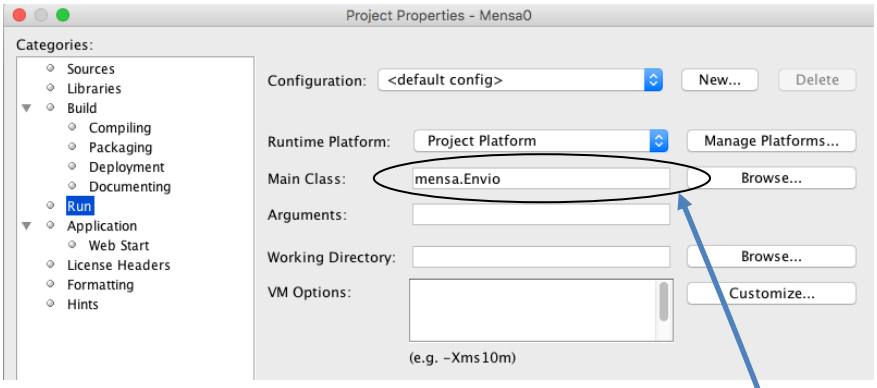
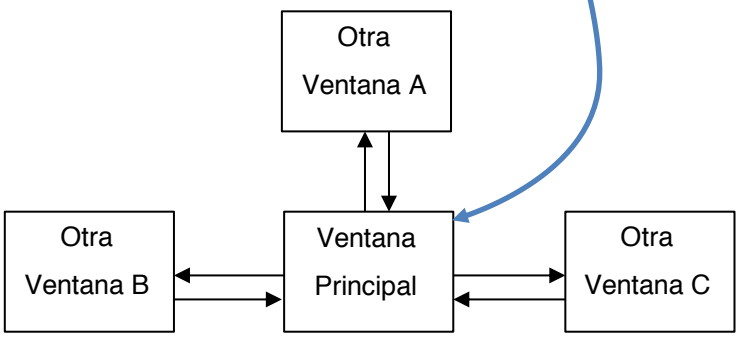
Práctica 2.
Varias ventanas

1 Arquitectura propuesta: estrella.

Se propone crear una arquitectura en estrella donde exista una ventana principal. Esta será la que se indique en las opciones de *NetBeans*:



En la clase asociada a la ventana principal se han de crear las referencias a todas las ventanas del proyecto. Las otras ventanas solo necesitan tener una referencia a la ventana principal: Gráficamente:

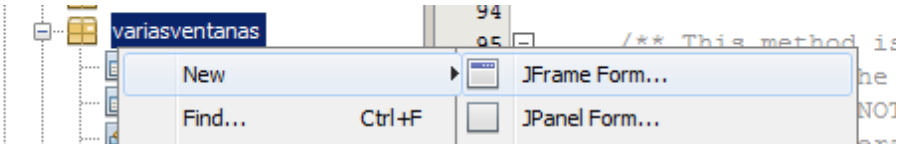


Nota: Cada arco indica que existe una referencia en la ventana/objeto origen que apunta a la destino.

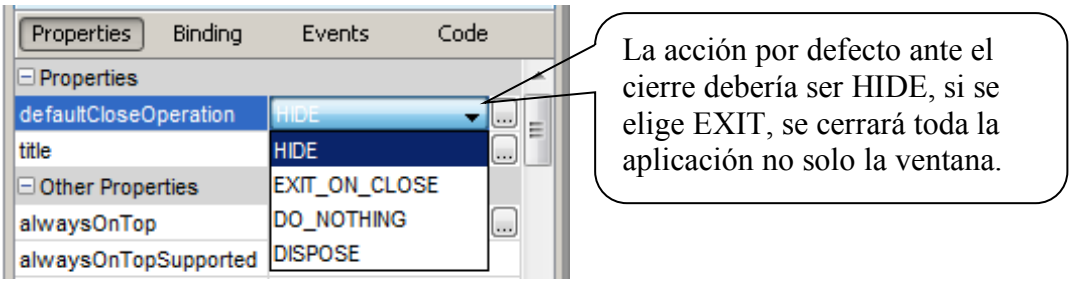
2 Crear una nueva ventana.

2.1 Diseñar la ventana.

Se crea una clase *jFrame* en el paquete de trabajo:



Modificar la opción defaultCloseOperation del *JFrame*:



Modificar la propiedad *tittle* para añadir el título de la ventana.

2.2 Crear el objeto asociado a la ventana.

Hay que seguir 2 pasos:

1. En la ventana principal: declarar una variable del tipo de la ventana.
2. En el constructor de la ventana principal: crear el objeto java llamando al constructor de la ventana.

```
NuevaVentana MiVentana; // Al final del código

// En constructor después de initComponents();
initComponents();
MiVentana=new NuevaVentana();
```

2.3 Utilizar la ventana.

Para hacer visible o invisible la ventana utilizar:

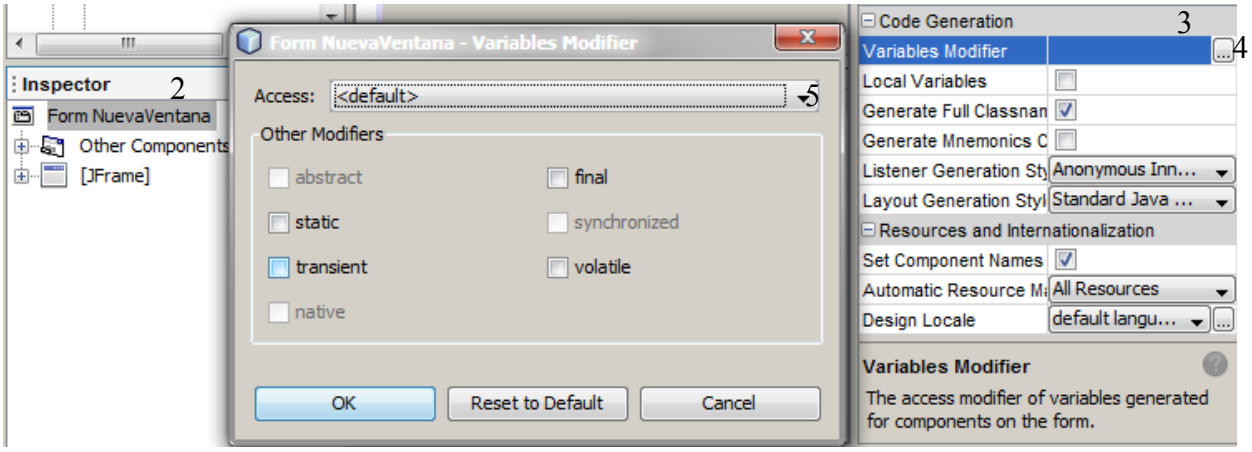
```
MiVentana.setVisible(boolean);
```

Actividad: crear una ventana principal que tenga un botón que lance otra ventana. Esta última tendrá un botón para hacerla invisible. Comprobar diferentes opciones del parámetro *defaultCloseOperation* de la nueva ventana.

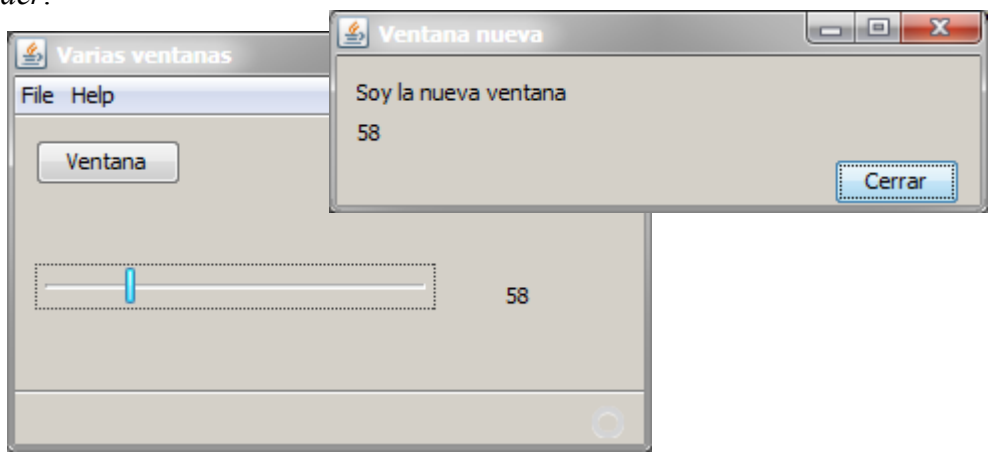
3 Acceso a objetos de otras ventanas.

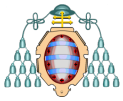
Acceder a los objetos de la Nueva Ventana desde la original, simplemente se utilizará la referencia que se declaró para crear la ventana. Pero por defecto las referencias que manejan los objetos gráficos de las ventanas son `private` con lo que no se puede acceder. Para cambiar este tipo de acceso desde *NetBeans*:

1. Ir a la Nueva Ventana.
2. Seleccionar en el inspector el objeto raíz (*root*) que será una *Form*.
3. En la ventana de propiedades aparecerá: *Code Generation*.
4. Lanzar el menú (pulsar sobre el botón de los 3 puntos) de *Variables Modifier*.
5. Cambiar el acceso a `<default>` si las dos ventanas están en el mismo paquete o `public` si están en paquetes distintos.



Actividad: crear en la ventana principal un *Slider* (`SlValor`) que varíe entre 0 y 255 (*mínimum, maximun*,) y que inicialmente esté en 128 (*value*). Crear una *Label* (`LValor`) que cuando cambie el valor del *Slider* (*StateChanged*) muestre el valor del *Slider*. Crear la *Label* `LVNValor` en la Ventana Nueva de manera que también muestre el valor del *Slider*.





Para que una ventana acceda a los objetos de la ventana que la creó necesita la referencia a la ventana. La manera típica de pasar esta referencia es mediante un constructor que admita como parámetro una referencia a dicha ventana. Ejemplo:

```
public class NuevaVentana extends javax.swing.JFrame {

    private VariasVentanasView VVV=null;
    public NuevaVentana(VariasVentanasView VVV)
    {
        this(); // Llamada al constructor sin parámetros
        this.VVV=VVV; // Asignar al campo el valor del parámetro
    }

    /** Creates new form NuevaVentana */
    public NuevaVentana() {
        initComponents();
    }
```

No eliminar este constructor.

En la ventana principal, al crear esta ventana usar el constructor que admite como parámetro una referencia a ventana indicando `this`.

```
public VariasVentanasView() {
    initComponents();

    NV=new NuevaVentana(this);
}
```

Actividad: crear en la Nueva Ventana tres botones: Min, Centrar y Max que modifiquen el *Slider* dándole valor respectivamente a 0, 128 y 255.

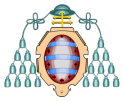
3.1 Métodos observadores.

Por razones de calidad de diseño del software suele ser conveniente crear métodos públicos `Tipo getX()` para acceder al objeto `X` de tipo `Tipo`.

4 Acceso a ventanas que no es la principal.

Si una ventana quiere acceder a una ventana que no es la principal tendrá que acceder a la ventana principal donde se encuentran las referencias al resto de ventanas y usando estas referencias podrá acceder a esas ventanas o a sus componentes.

Las referencias a la ventanas en la ventana principal tendrán que ser accesibles (*public* , *package*, métodos *get*) al resto de ventanas.



5 Aparición de una ventana antes que la principal.

Para conseguir que se cree una ventana antes que la principal hay que conseguir que la principal no esté inicialmente visible, para ello modificar su función `main` poniendo a *false* opción de visible.

```
java.awt.EventQueue.invokeLater(new Runnable() {
    public void run() {
        new Envio().setVisible(false);
    } });
```

La ventana inicial tiene que ser creada desde la principal (como todas) y tiene que estar inicialmente visible, esto puede lograrse en la ventana inicial tras crearla o en el constructor de la Inicial. La ventana inicial tendrá como parámetro la ventana inicial pues en algún momento tendrá que poner visible la ventana principal.