EJERCICIO GUIADO. JAVA: POO. HERENCIA. DIALOGOS PROPIOS

Reutilización de código

La gran ventaja de la Herencia es la posibilidad de aprovechar Clases ya creadas (bien sea por nosotros mismos o por otros programadores) para crear nuevas Clases. De esta forma, no tenemos que crear la nueva Clase desde el principio, y solo tenemos que añadir algunos cambios a la Clase original, sin que esta se vea afectada.

De esta manera, podemos tomar Clases de las librerías de Java y crear a partir de ellas nuevas Clases para nuestras necesidades específicas. Estas clases luego pueden ser incluidas fácilmente en futuros proyectos.

Cuadro de Diálogo Propio

Como ejemplo de todo esto, crearemos un cuadro de diálogo propio que puede ser usado muy a menudo en nuestros proyectos. Concretamente, será un cuadro de diálogo que nos permita introducir una fecha (dia / mes / año)

Debes tener en cuenta que los cuadros de diálogo son objetos de la clase JDialog, por lo que este cuadro de diálogo propio tendrá que derivar (heredar) de la clase JDialog.

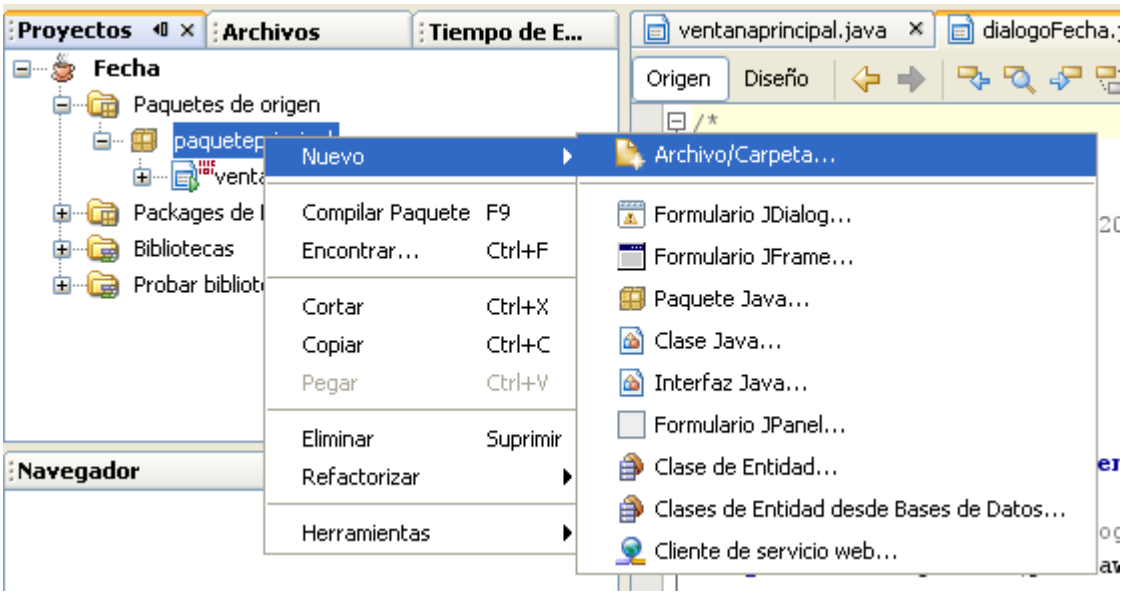
La clase JDialog se comporta como una ventana (un JFrame) así que su programación es relativamente sencilla.

Veremos el caso concreto de programar un Cuadro de Diálogo Propio a través del ejercicio guiado que viene a continuación.

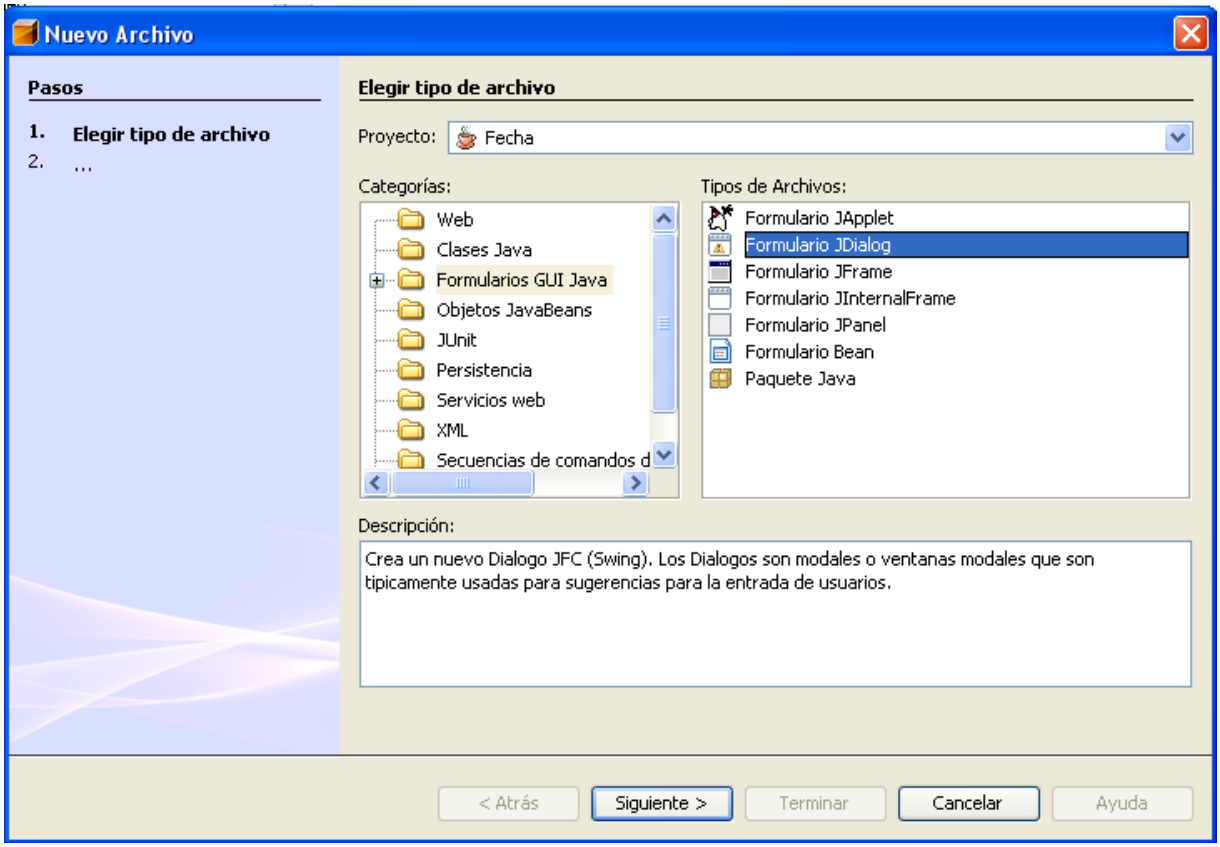
EJERCICIO GUIADO

1. Crea un proyecto llamado *Fecha*, que tenga un paquete principal llamado *paqueteprincipal* y un JFrame llamado *ventanaprincipal*:

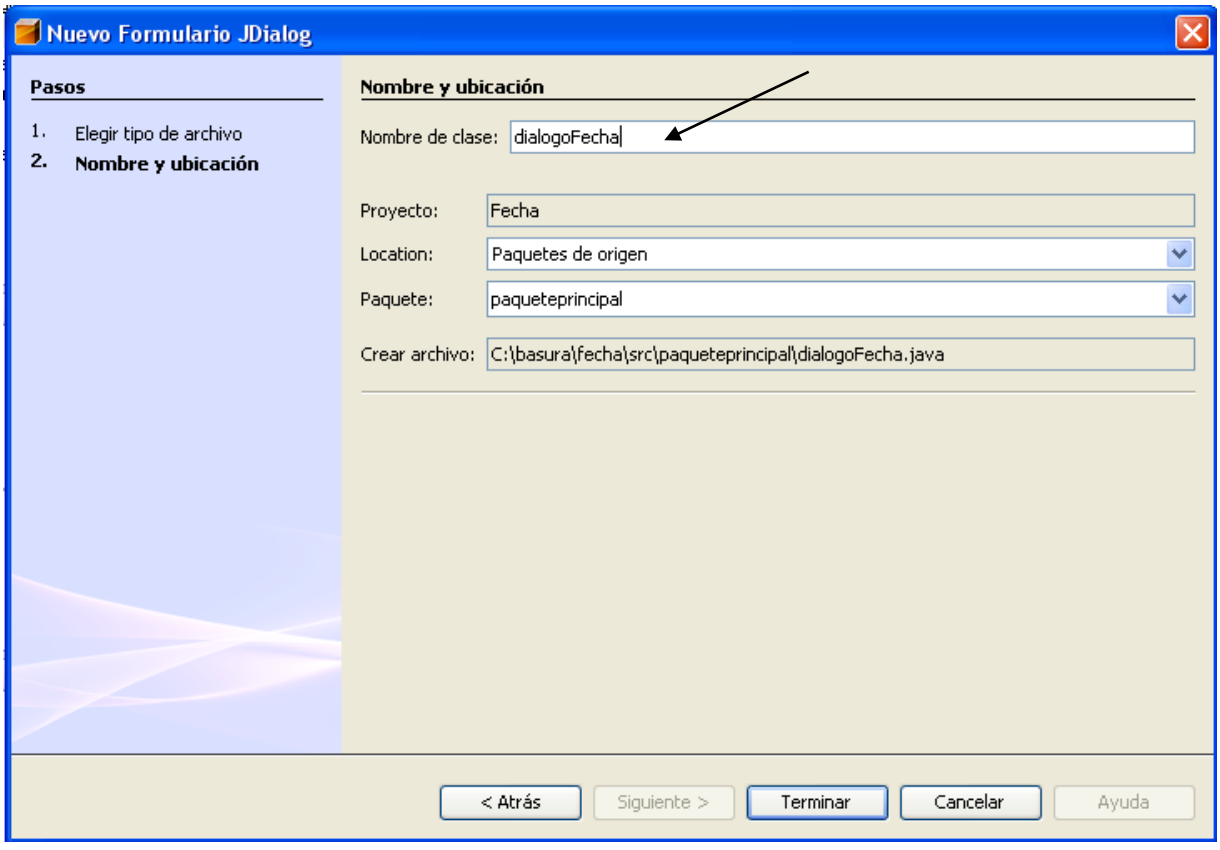
2. Para crear un cuadro de diálogo propio, debes hacer clic con el botón derecho del ratón sobre el paquete principal y activar la opción *Nuevo – Archivo/Carpeta.* A través de esta opción accedemos a un menú desde donde podemos añadir a nuestro proyecto distintos tipos de clases.



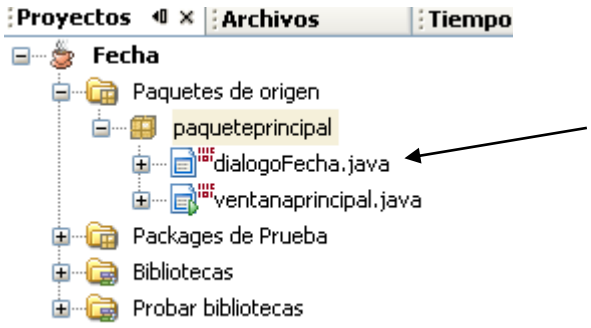
3. En la ventana que aparece, escogeremos en la parte izquierda la opción *Formularios GUI Java,* y en la parte derecha escogeremos la opción *Formulario JDialog*. Luego activa el botón *Siguiente*.



4. A continuación tendremos que indicar el nombre que tendrá la clase correspondiente a nuestro cuadro de diálogo propio. En nuestro caso llamaremos a esta clase *dialogoFecha*.



5. Y pulsa el botón *Terminar*. Observarás que se ha creado una nueva Clase dentro del proyecto llamada *dialogoFecha*.



6. Es interesante que veas el código de esta clase. Haz doble clic sobre ella:

```
public class dialogoFecha extends javax.swing.JDialog {

    /** Creates new form dialogoFecha */
    public dialogoFecha(java.awt.Frame parent, boolean modal) {
        super(parent, modal);
        initComponents();
    }
```

Si observas el código verás que es muy similar al de las Clases JFrame, es decir, a la de la clase de la ventana principal.

Debes observar que la clase *dialogoFecha* que vas a crear es heredada de JDialog, y también verás que hay un constructor similar al de los JFrame, aunque con parámetros:

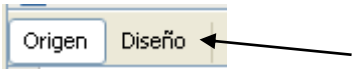
```
public class dialogoFecha extends javax.swing.JDialog {

    /** Creates new form dialogoFecha */
    public dialogoFecha(java.awt.Frame parent, boolean modal) {
        super(parent, modal);
        initComponents();
    }
```

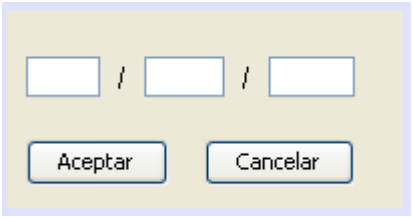
Hereda de JDialog

Constructor

7. Además, tenemos la ventaja de poder usar la ventana de diseño para crear nuestro cuadro de diálogo *dialogoFecha*:



8. Usaremos la ventana de diseño para darle forma a nuestro cuadro de diálogo. Recuerda que la finalidad de este cuadro de diálogo será la de permitirle al usuario introducir una fecha. Diséñalo para que quede así:



9. Asigna nombre a cada elemento:

    - El cuadro de texto para el día se llamará: txtDia
    - El cuadro de texto para el mes se llamará: txtMes
    - El cuadro de texto para el año se llamará: txtAnio
    - El botón Aceptar se llamará btnAceptar
    - El botón Cancelar se llamará btnCancelar

10. Como ves, hasta ahora la creación de un cuadro de diálogo propio es algo muy sencillo, ya que es muy similar a la creación de una ventana. En este proceso (al igual que en la creación de una ventana) participa el concepto de herencia.

11. Ahora hay que dotar a nuestro cuadro de diálogo de las propiedades y los métodos necesarios para que sea fácil de usar. Antes de programar todo esto, aquí tienes una descripción de lo que queremos añadir al cuadro de diálogo:

Propiedades:

Dia - entero
Mes - entero
Anio - entero
BotonPulsado - entero

Métodos:

getFecha()

- Este método devolverá una cadena (String) con la fecha en este formato:
  dia/mes/año

getFechaLarga()

- Este método devolverá una cadena con la fecha en el siguiente formato:
  Dia de Mes(en letras) de Año

getBotonPulsado()

- Este método devolverá el valor de la propiedad BotonPulsado. Esta propiedad podrá tener un 0 o un 1. Tendrá un 0 si se pulsa el botón Aceptar, y un 1 si se pulsa el botón Cancelar.

12. Bien, empecemos. Programar las propiedades es algo sencillo, ya que son simplemente variables globales a la clase *dialogoFecha*:

```
public class dialogoFecha extends javax.swing.JDialog {

    int Dia;
    int Mes;
    int Anio;
    int BotonPulsado;
```

Propiedades de la clase dialogoFecha

13. Cuando el usuario pulse el botón Cancelar, el cuadro de diálogo de fecha debe introducir el valor 1 dentro de su propiedad BotonPulsado, y el cuadro de diálogo se cerrará. Para programar esto, acude a la ventana de diseño y entra en el *actionPerformed* del botón *btnCancelar*.

```
    private void btnCancelarActionPerformed(java.awt.event.ActionEvent evt) {
// TODO: Agrege su codigo aqui:
        BotonPulsado=1;
        this.dispose();
    }
```

Como puedes observar en el código, asigno el 1 a la propiedad BotonPulsado y luego uso el método *dispose* para cerrar el cuadro de diálogo.

14. Cuando el usuario pulse el botón Aceptar, el cuadro de diálogo de fecha debe introducir el valor de los JTextField txtDia, txtMes y txtAnio en sus respectivas propiedades: Dia, Mes y Anio. Luego, se debe introducir un 0 en la propiedad BotonPulsado, y finalmente se cerrará el cuadro de diálogo.

    Accede al *actionPerformed* del botón btnAceptar y programa lo siguiente…

```
    private void btnAceptarActionPerformed(java.awt.event.ActionEvent evt) {
// TODO: Agrege su codigo aqui:
        Dia = Integer.parseInt(txtDia.getText());
        Mes = Integer.parseInt(txtMes.getText());
        Anio = Integer.parseInt(txtAnio.getText());

        BotonPulsado=0;
        this.dispose();
    }
```

15. Ya hemos programado las acciones a realizar cuando se pulse el botón Aceptar y Cancelar. Ahora programaremos el resto de los métodos. Empezaremos por el método *getFecha*.

    El método *getFecha* devolverá una cadena con la fecha en formato dia/mes/año. Por tanto, este método tendrá la siguiente forma. Debes escribir este código dentro de la clase *dialogoFecha*. El mejor sitio puede ser debajo del constructor.

```
public String getFecha() {
    String cadena;

    cadena = Dia+"/"+Mes+"/"+Anio;
    return cadena;
}
```

Observa el código. Se unen las propiedades dia, mes y año en una cadena, separadas por el símbolo “/”. Luego se devuelve la cadena.

16. Ahora programaremos el método *getFechaCompleta*. Este método devuelve la fecha con formato largo, usando el mes en letras. El método *getFechaCompleta* tendrá el siguiente aspecto. Puedes programarlo a continuación del método del punto anterior:

```
public String getFechaCompleta() {
    String cadena;
    cadena = Dia + " de ";
    if (Mes==1) {
        cadena = cadena + "Enero";
    } else if (Mes==2) {
        cadena = cadena + "Febrero";
    } else if (Mes==3) {
        cadena = cadena + "Marzo";
    } else if (Mes==4) {
        cadena = cadena + "Abril";
    } else if (Mes==5) {
        cadena = cadena + "Mayo";
    } else if (Mes==6) {
        cadena = cadena + "Junio";
    } else if (Mes==7) {
        cadena = cadena + "Julio";
    } else if (Mes==8) {
        cadena = cadena + "Agosto";
    } else if (Mes==9) {
        cadena = cadena + "Septiembre";
    } else if (Mes==10) {
        cadena = cadena + "Octubre";
    } else if (Mes==11) {
        cadena = cadena + "Noviembre";
    } else if (Mes==12) {
        cadena = cadena + "Diciembre";
    }
    cadena = cadena + " de "+Anio;
    return cadena;
}
```

Como puedes observar, se concatena el día en una cadena y luego, dependiendo del valor de la propiedad Mes, se concatena un mes en letras. Finalmente se concatena la propiedad Anio y se devuelve la cadena.

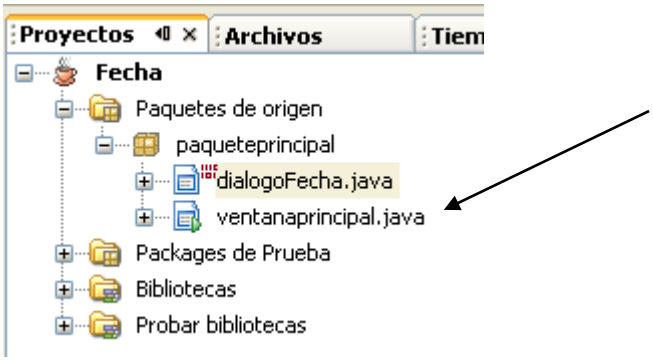
17. Y ya para terminar con la programación de nuestro *dialogoFecha*, programaremos el método *getBotonPulsado*, que básicamente devuelve la propiedad *BotonPulsado*, la cual contiene un

0 si se pulsó Aceptar y un 1 si se pulsó Cancelar. La programación de este método es muy sencilla:

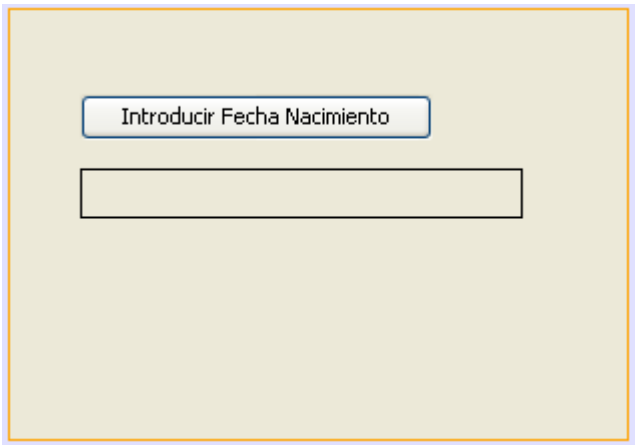
```
public int getBotonPulsado() {
    return BotonPulsado;
}
```

La clase *dialogoFecha* está terminada. Gracias a ella podremos crear cuadros de diálogo en nuestros proyectos para introducir fechas. Para probar esta clase, usaremos nuestra ventana principal.

18. Acude a la ventana principal, haciendo doble clic sobre ella en la ventana de proyectos:



19. Diseña esta ventana para que tenga la siguiente forma:



El botón se llamará *btnFechaNacimiento* y la etiqueta con borde se llamará *etiFechaNacimiento*.

20. El programa funcionará de la siguiente forma:

    - Al pulsar el botón *Introducir Fecha Nacimiento* aparecerá un cuadro de diálogo del tipo *dialogoFecha*, donde el usuario introducirá la fecha de nacimiento.
    - Al pulsar Aceptar, dicha fecha aparecerá en la etiqueta *etiFechaNacimiento*.
    - Al pulsar Cancelar, la etiqueta *etiFechaNacimiento* se borrará.

21. Nuestro programa necesitará un cuadro de diálogo del tipo *dialogoFecha*. Este cuadro lo declararemos en la zona de variables globales de la ventana principal y lo llamaremos *nacimiento*…

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

    dialogoFecha nacimiento;

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

22. El objeto *nacimiento* es del tipo *dialogoFecha*. Recuerda que los objetos que declaras de forma global, luego debes construirlos en el constructor. Así pues construiremos el objeto *nacimiento*.

    Para construir un objeto que sea un cuadro de diálogo hay que añadir dos parámetros: null, y luego indicar true o false, según quieras que el cuadro de dialogo sea modal o no. En nuestro caso, queremos que el cuadro de diálogo *nacimiento* sea modal, por lo que indicaremos true:

```
/** Creates new form ventanaprincipal */
public ventanaprincipal() {
    initComponents();
    nacimiento = new dialogoFecha(null,true);
}
```

Se construye el cuadro de diálogo *nacimiento*. Será modal (true)

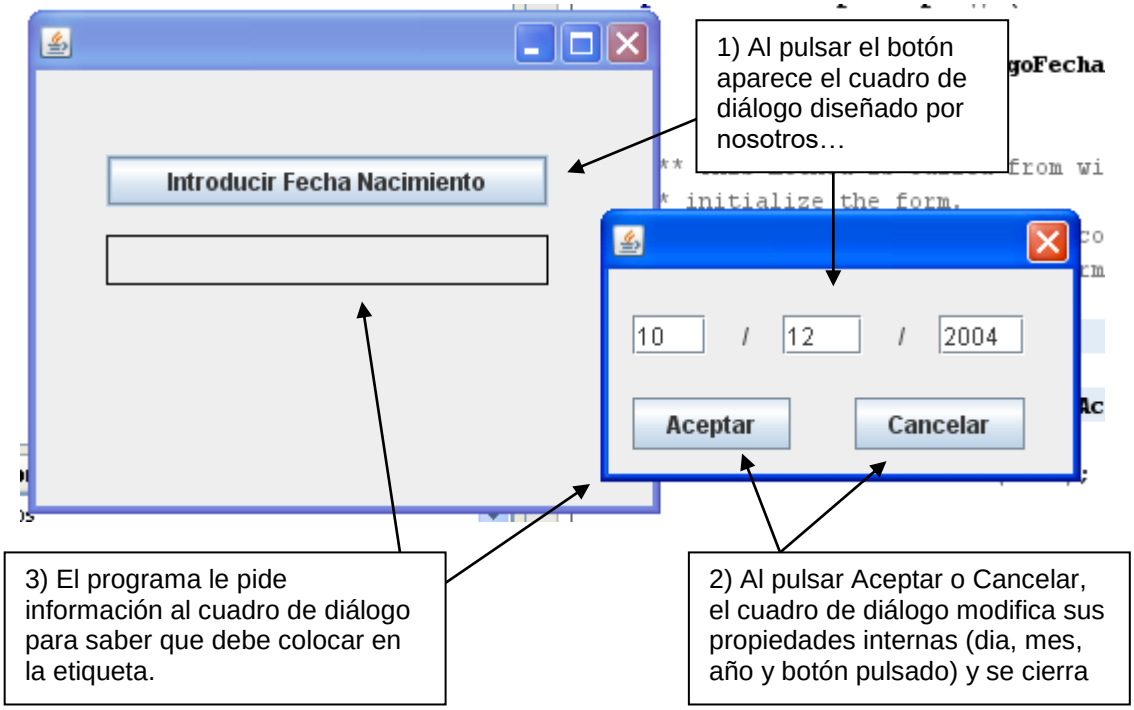
23. Programa el *actionPerformed* del botón *btnFechaNacimiento*. Este botón mostrará al cuadro de diálogo *nacimiento*, donde el usuario introducirá una fecha. Luego, comprobaremos si el usuario pulsó Aceptar o Cancelar. En el caso de que pulsara Aceptar, el programa introducirá en la etiqueta *etiFechaNacimiento* la fecha escrita por el usuario. Todo esto se hace preguntando al objeto *nacimiento* a través de los métodos programados. Observa el código:

```
    private void btnFechaNacimientoActionPerformed(java.awt.event.ActionEvent evt) {
// TODO: Agrege su codigo aqui:
        nacimiento.setVisible(true); //muestro el cuadro de diálogo

        if (nacimiento.getBotonPulsado()==0) { //si pulso Aceptar
            etiFechaNacimiento.setText(nacimiento.getFecha());
        } else { //si pulsó Cancelar
            etiFechaNacimiento.setText("");
        }

    }
```

24. Ejecuta el programa y observa el funcionamiento del botón *btnFechaNacimiento* y del cuadro de diálogo *nacimiento*



EJERCICIO

Añade otra etiqueta a la ventana principal. Modifica el código del botón *btnFechaNacimiento* de forma que también aparezca en esta etiqueta la fecha elegida por el usuario en formato largo. Esto debe suceder solo si el usuario Acepta el cuadro de diálogo. En caso de que el usuario cancele el cuadro de diálogo esta etiqueta se borrará.

CONCLUSIÓN

Es posible crear cuadros de diálogos propios, con las características que más nos interesen.

Estos cuadros de diálogo son clases que derivan de la clase JDialog.

Los cuadros de diálogo se comportan y manejan básicamente igual que las ventanas.

La gran ventaja de crear cuadros de diálogos propios, es que luego se podrán usar en otros proyectos.