UNIVERSIDAD DE CÓRDOBA

FACULTAD DE INGENIERÍAS

PROGRAMA INGENIERÍA DE SISTEMAS

CURSO PROGRAMACION III

Ejemplo realización en UML e Interfaces en Java

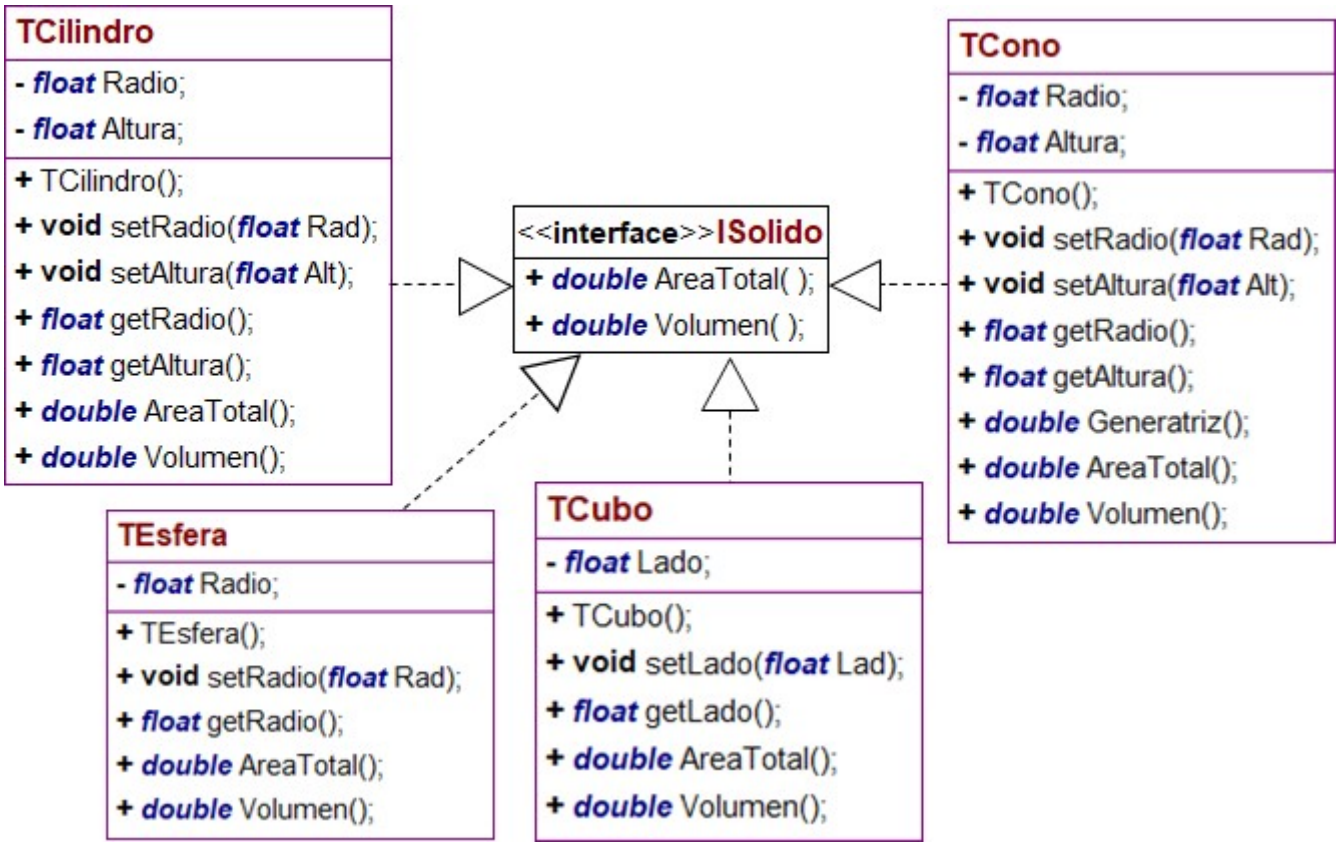
LUIS OLASCOAGA SURMAY

MONTERÍA AGOSTO DE 2020

1. Presentación del ejemplo

En este ejemplo de aplicación de interfaces y de relación de realización en UML, presentamos el diseño UML para clases que implementan (realizan) una misma interface, cuyos métodos permiten calcular el área total y el volumen de objetos o figuras solidas; es decir, de tres dimensiones, de tal suerte que la interface será implementada por cuatro figuras solidas a saber: Cubo, cono, esfera y cilindro; considerando además una clase independiente para cada una de estas figuras y que no existen un ancestro común entre las mismas; o sea , que no se va a usar relaciones de herencia entre dichas clases. Por otra parte, el diseño UML será implementado usando el lenguaje de programación Java, mediante una aplicación o proyecto de ventana desarrollado en *NetBeans*.

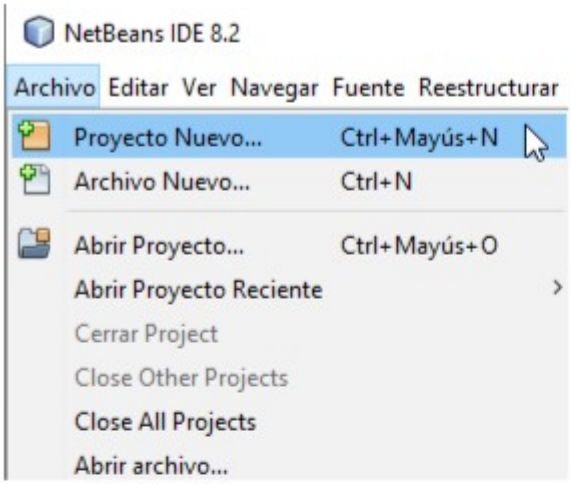
2. Diseño UML



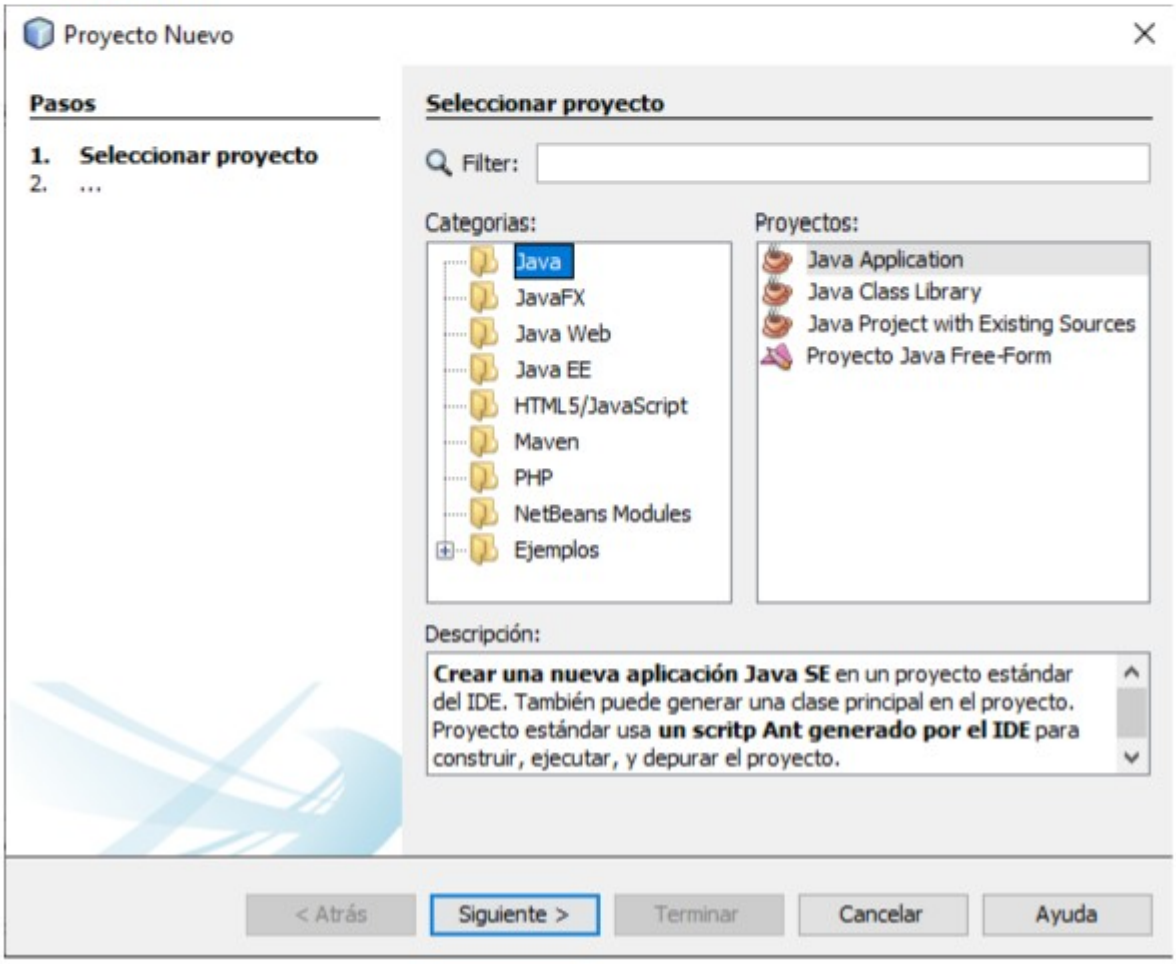
Luis Roberto Olascoaga Surmay

3. Creación del proyecto en NetBeans

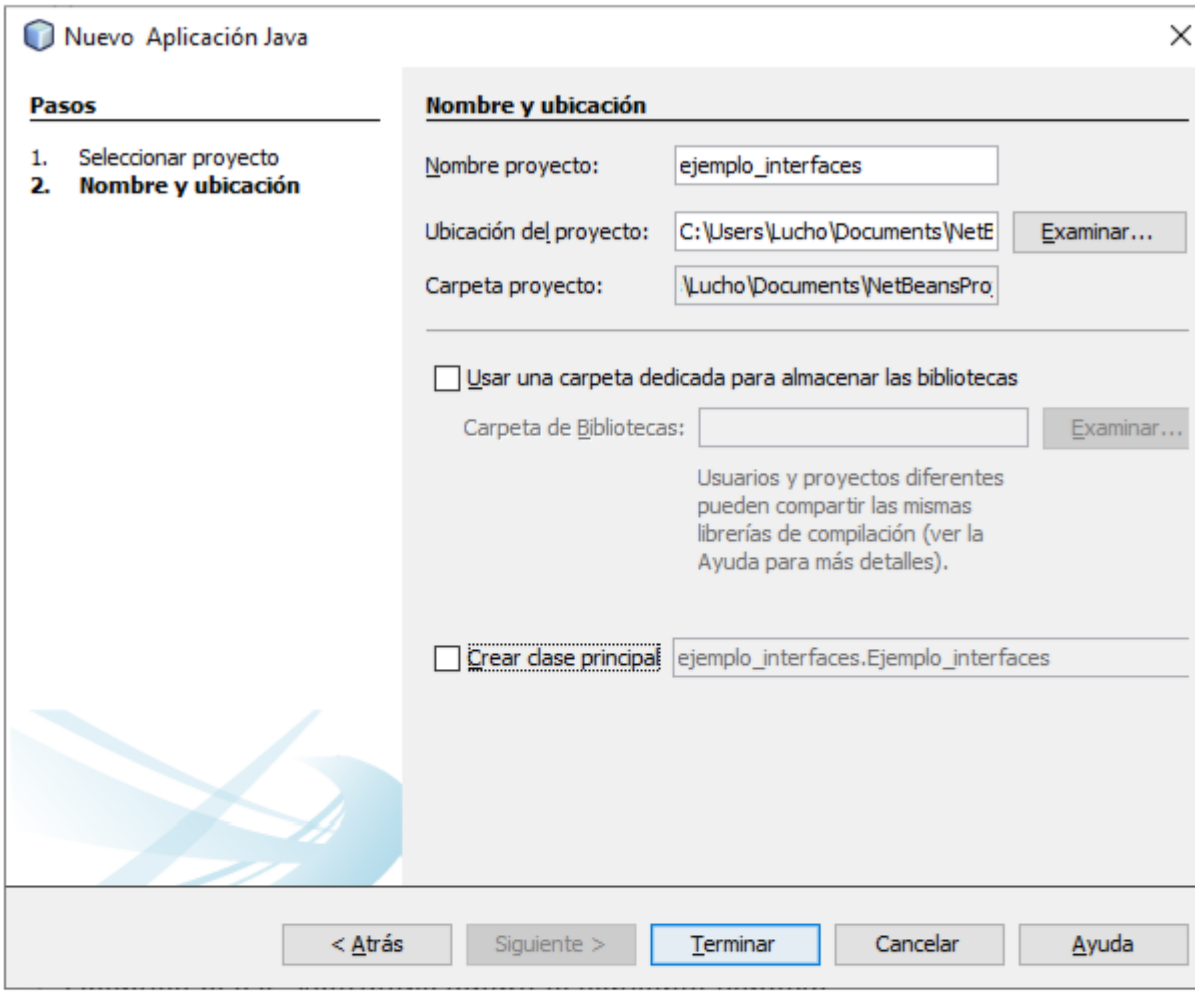
- Haga click en la opción de menú ***Archivo*** y escoja la opción ***Proyecto Nuevo***:



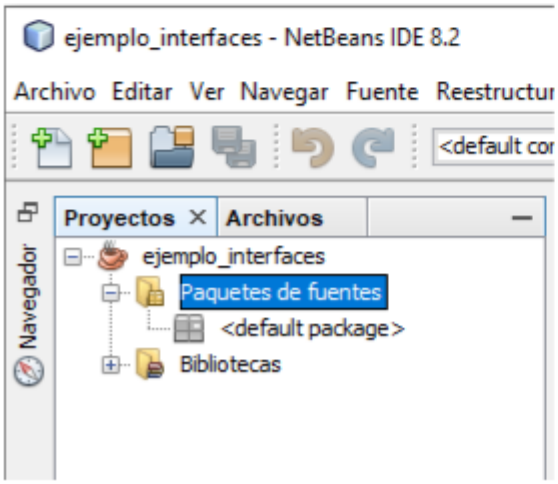
- En esta ventana en el panel "***Categorias***" escoja el nodo "***Java***", en el panel "***Proyectos***" seleccione el ítem "***Java Application***" y haga click en el botón "***Siguiente***".



- Ahora verá la siguiente ventana, en la cual en la entrada "***Nombre Proyecto***" escriba ***ejemplo_interfaces*** y desmarque la casilla de verificación "***Crear Clase Principal***; después haga click en el botón ***Terminar***.



- Después el IDE *NetBeans* tendrá el siguiente aspecto:

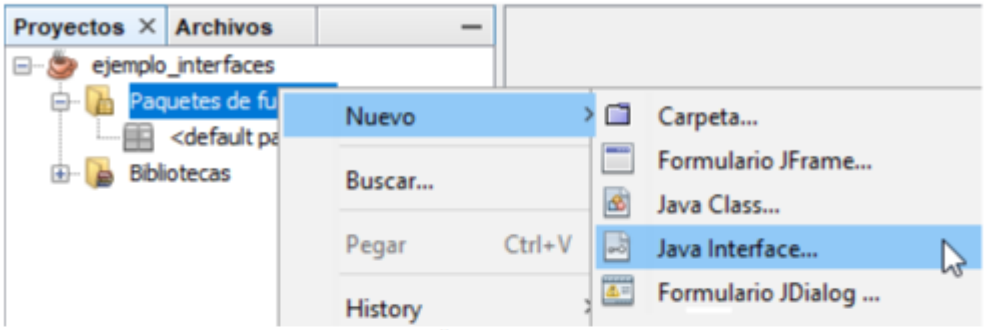


Luis Roberto Olascoaga Surmay

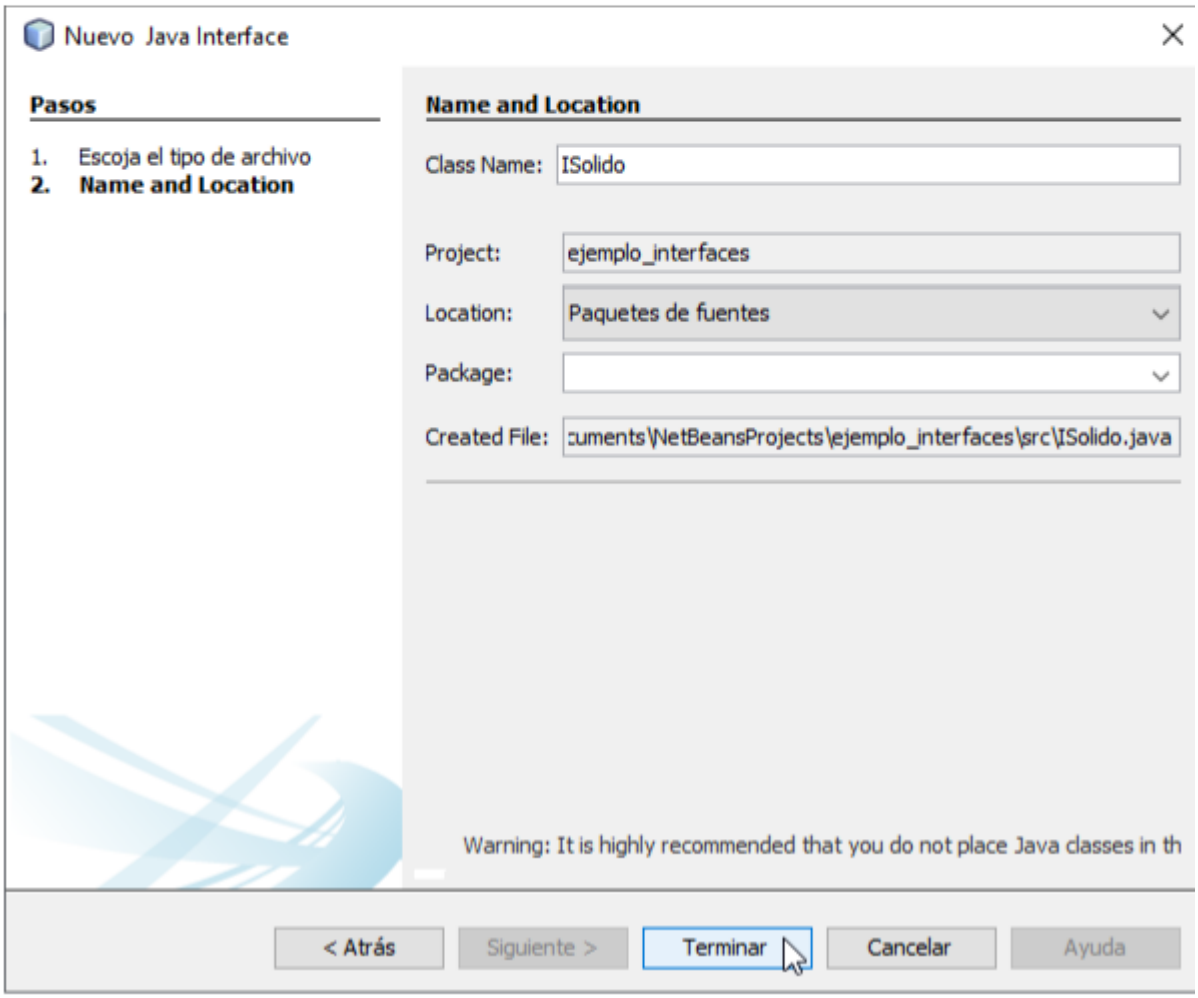
4. Implementación clases diagrama UML

a) Creación interface *ISolido*

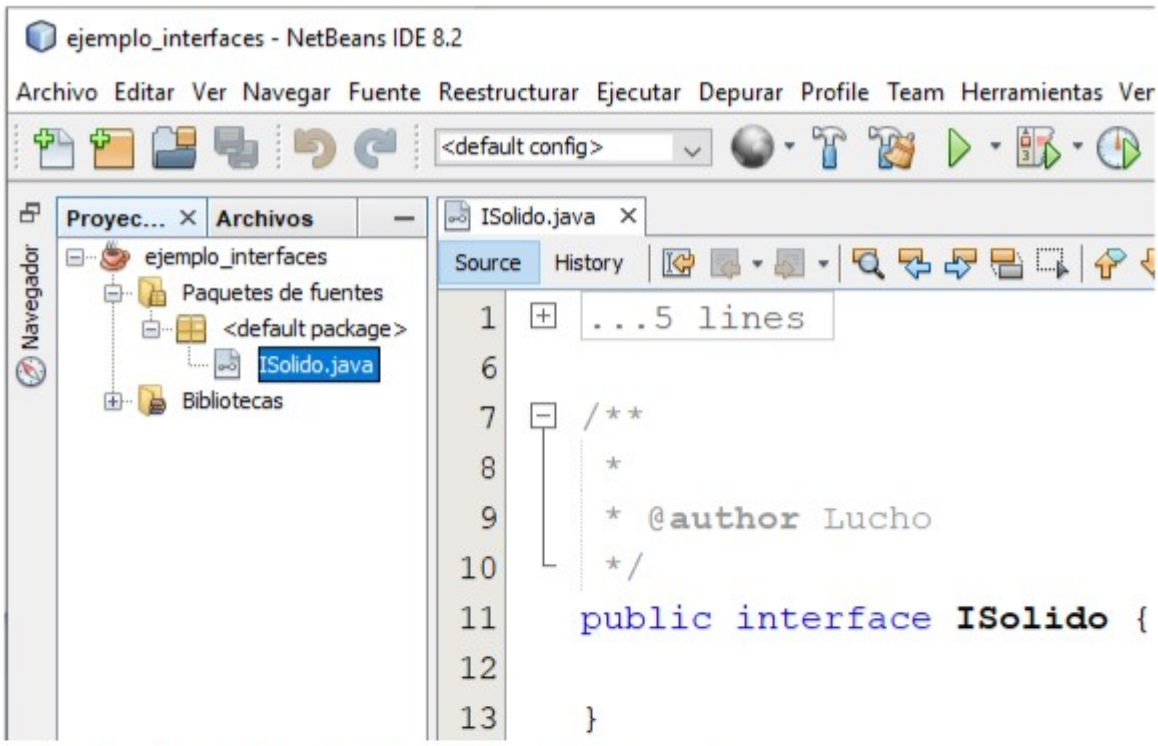
- Para crear esta nueva clase en el proyecto, haga click derecho sobre el nodo ***paquetes de fuentes*** del proyecto ***ejemplo_interfaces***, seleccione la opción ***Nuevo*** y después ***Java Interface…***, tal como se muestra abajo:



- En esta ventana en la entrada "***Class Name***" ponga ***ISolido*** como nombre de la clase y haga click en el botón ***Terminar***.



- En este punto el IDE se vera de la siguiente manera

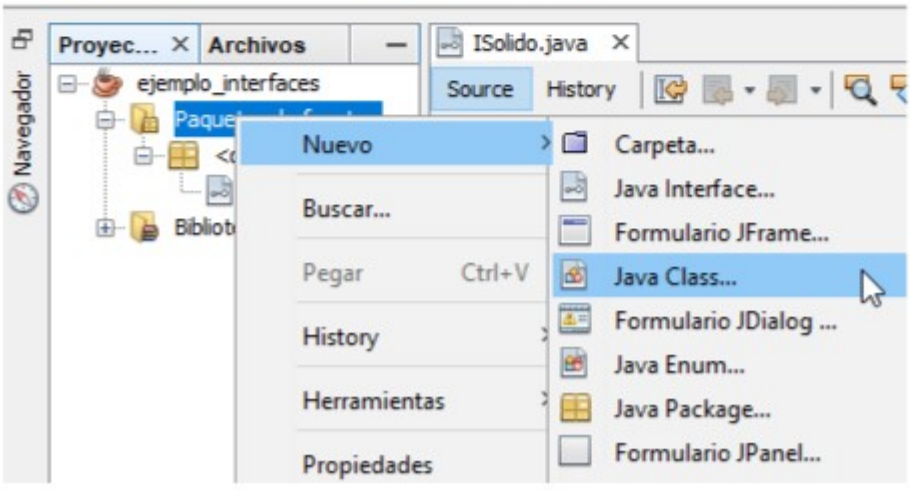


- Defina la interface ***ISolido*** (archivo *ISolido*.java) como sigue:

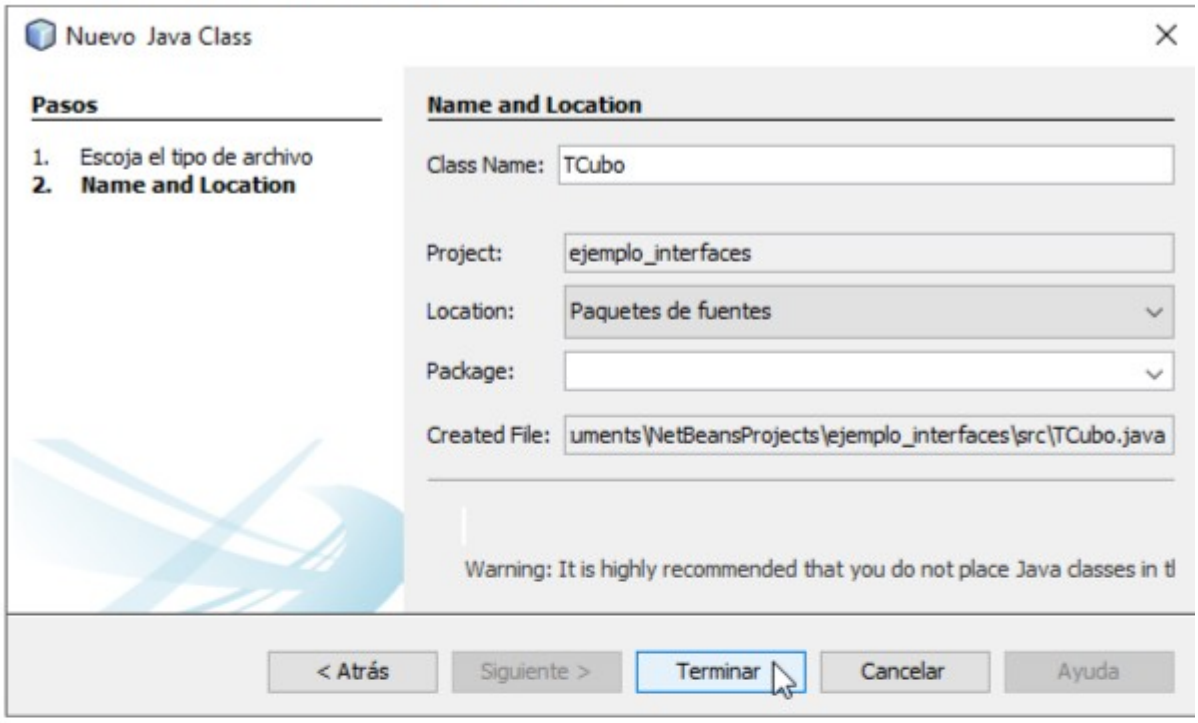
```
1 public interface ISolido {
2   public double AreaTotal();
3   double Volumen();
4 }
```

b) Creación clase *TCubo*

- Para crear esta nueva clase en el proyecto, haga click derecho sobre el nodo ***paquetes de fuentes*** del proyecto ***ejemplo_interfaces***, seleccione la opción ***Nuevo*** y después ***Java Class...***, tal como se muestra abajo:



- En esta ventana en la entrada "***Class Name***" ponga ***TCubo*** como nombre de la clase y haga click en el botón ***Terminar***.



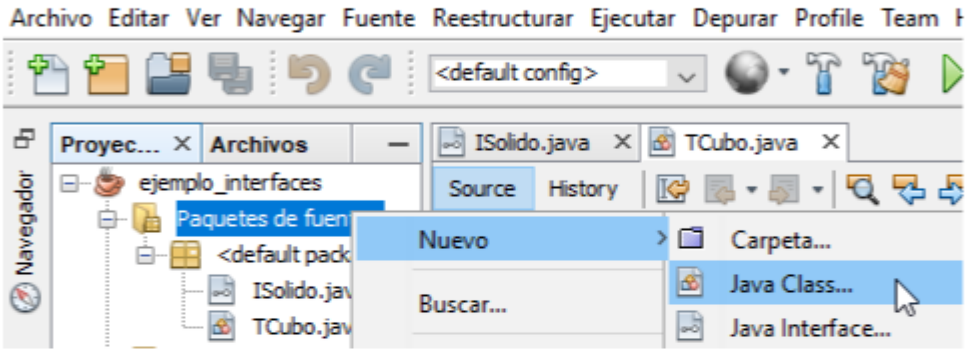
- Implemente la clase ***TCubo*** (archivo *TCubo*.java) como sigue:

```
1  public class TCubo implements ISolido {
2
3   private float Lado;
4
5   public TCubo(){
6    Lado=0;
7   }
8
9   public void setLado(float Lad){
10   if(Lad>=0){
11    Lado=Lad;
12   }
13  }
14
15  public float getLado(){
16   return Lado;
17  }
18
19  @Override
20  public double AreaTotal(){
21   return 6*Lado*Lado;
22  }
```

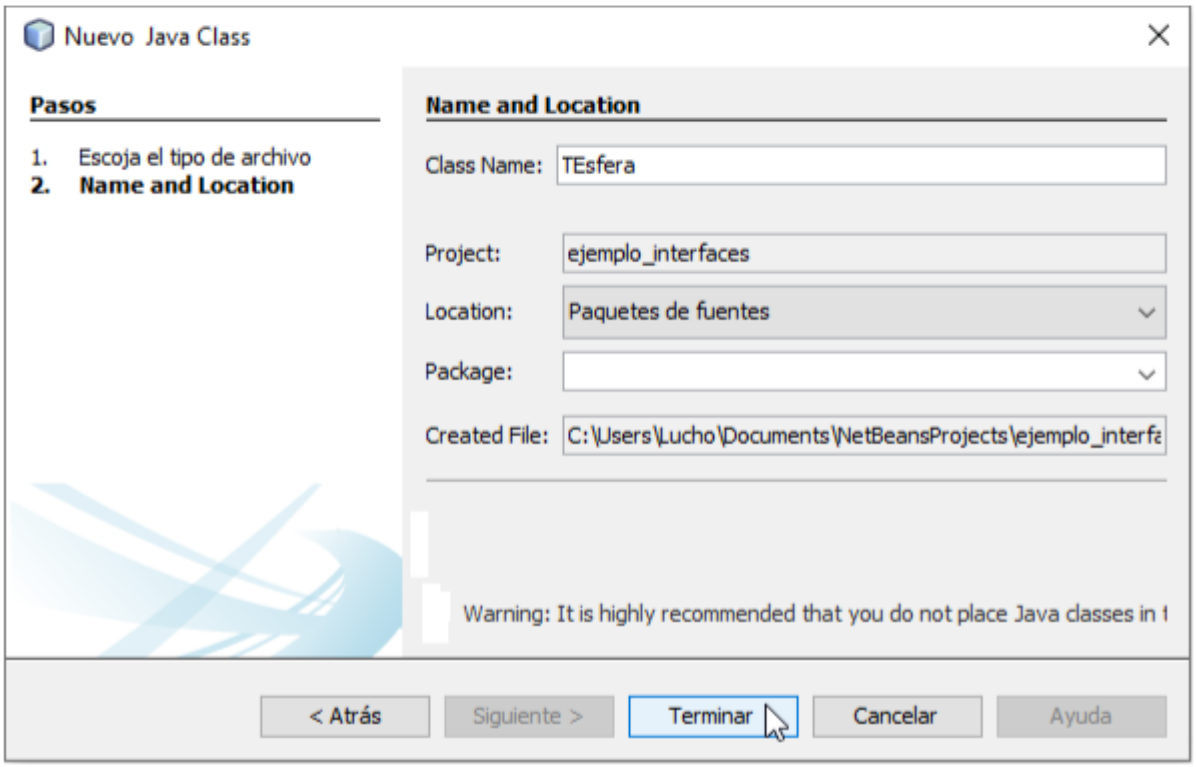
```
23
24  @Override
25  public double Volumen(){
26    return Lado*Lado*Lado;
27  }
28
29 }
```

c) Creación clase *TEsfera*

- Para crear esta nueva clase en el proyecto, haga click derecho sobre el nodo ***paquetes de fuentes*** del proyecto ***ejemplo_interfaces***, seleccione la opción ***Nuevo*** y después ***Java Class...***, tal como se muestra abajo:



- En esta ventana desplegada ingrese ***TEsfera*** *como* nombre de la clase en la entrada titulada como "***Class Name***", luego haga click en el botón ***Terminar***.



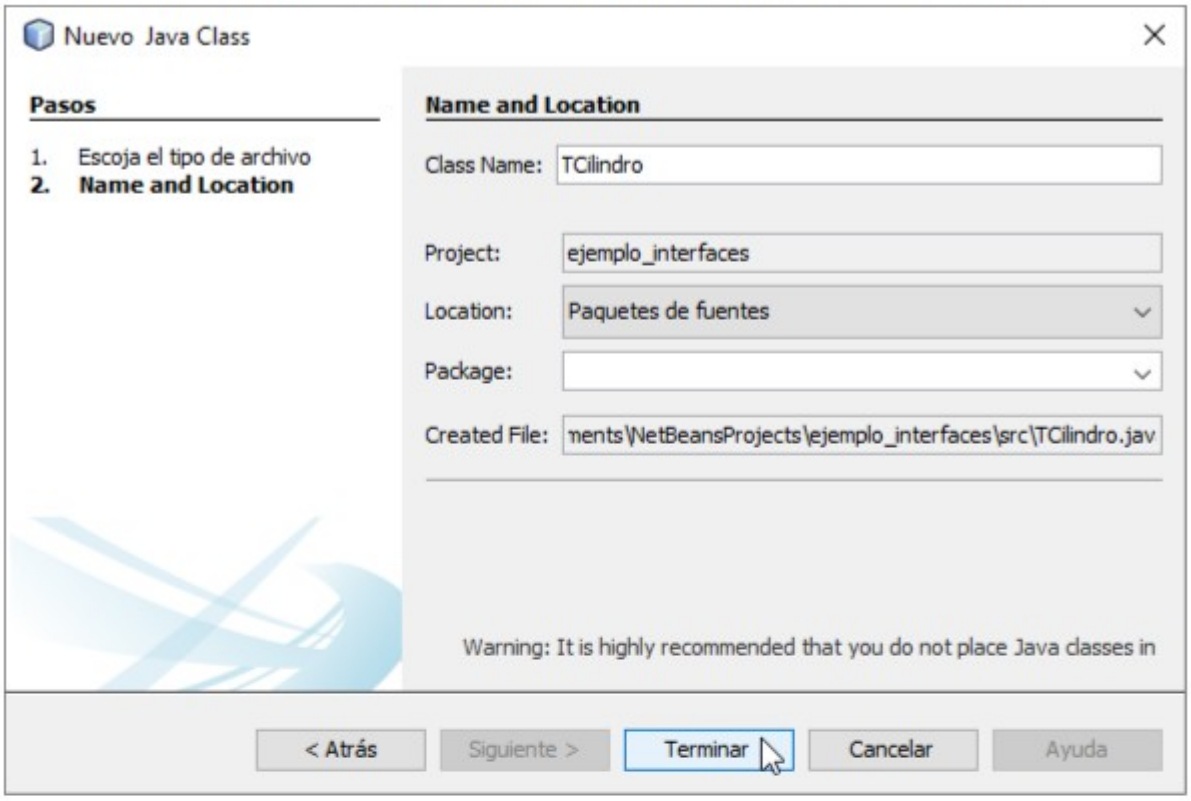
- Implemente la clase ***TEsfera*** (archivo *TEsfera*.java) como sigue:

```
1  public class TEsfera implements ISolido {
2
3   private float Radio;
4
5   public TEsfera(){
6    Radio=0;
7   }
8
9   public void setRadio(float Rad){
10   if(Rad>=0){
11    Radio=Rad;
12   }
13  }
14
15  public float getRadio(){
16   return Radio;
17  }
18
19  @Override
20  public double AreaTotal(){
21   return 4*Math.PI*Radio*Radio;
22  }
23
24  @Override
25  public double Volumen(){
26   return 4*Math.PI*Radio*Radio*Radio/3;
27  }
28
29 }
```

d) Creación clase *TCilindro*

- Para crear esta nueva clase en el proyecto, haga click derecho sobre el nodo ***paquetes de fuentes*** del proyecto ***ejemplo_interfaces***, seleccione la opción ***Nuevo*** y después ***Java Class...***, tal como se muestra abajo:

- En la ventana que se presenta en la entrada "***Class Name***", ponga ***TCilindro*** *como* nombre de la clase y después haga click en el botón ***Terminar***.



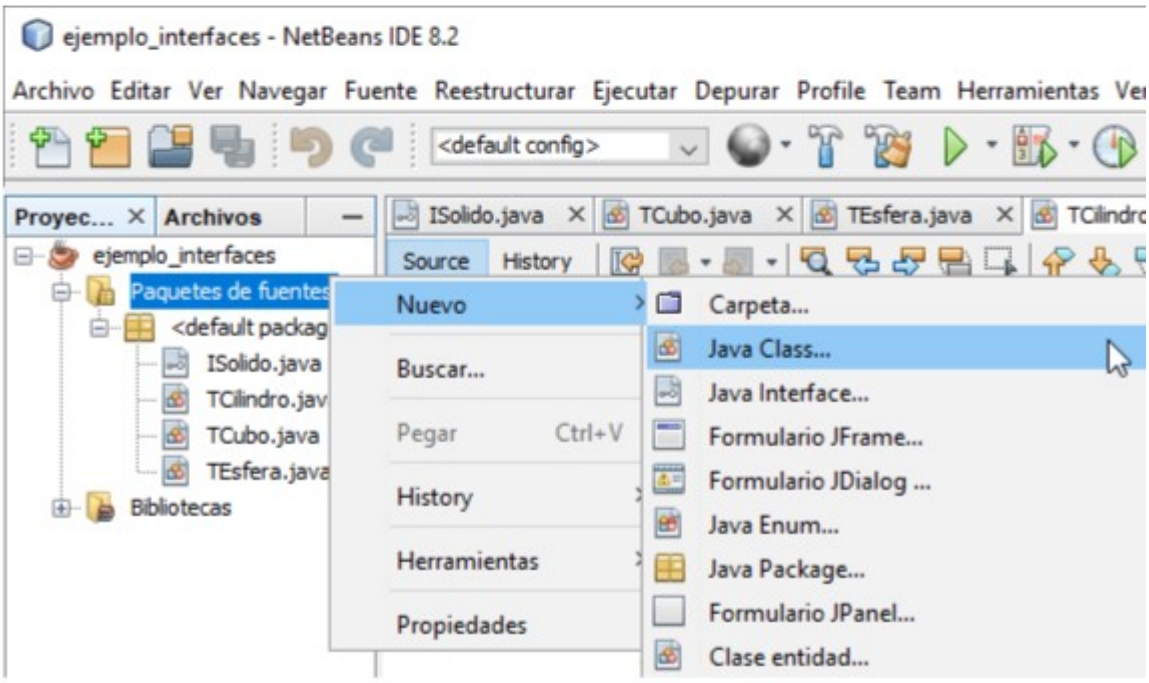
- Implemente la clase ***TCilindro*** (archivo *TCilindro*.java) como sigue:

```
1  public class TCilindro implements ISolido {
2
3    private float Radio;
4    private float Altura;
5
6    public TCilindro(){
7      Radio=0;
8      Altura=0;
9    }
10
11   public void setRadio(float Rad){
12     if(Rad>=0){
13       Radio=Rad;
14     }
15   }
16
17   public void setAltura(float Alt){
18     if(Alt>=0){
19       Altura=Alt;
20     }
21   }
```

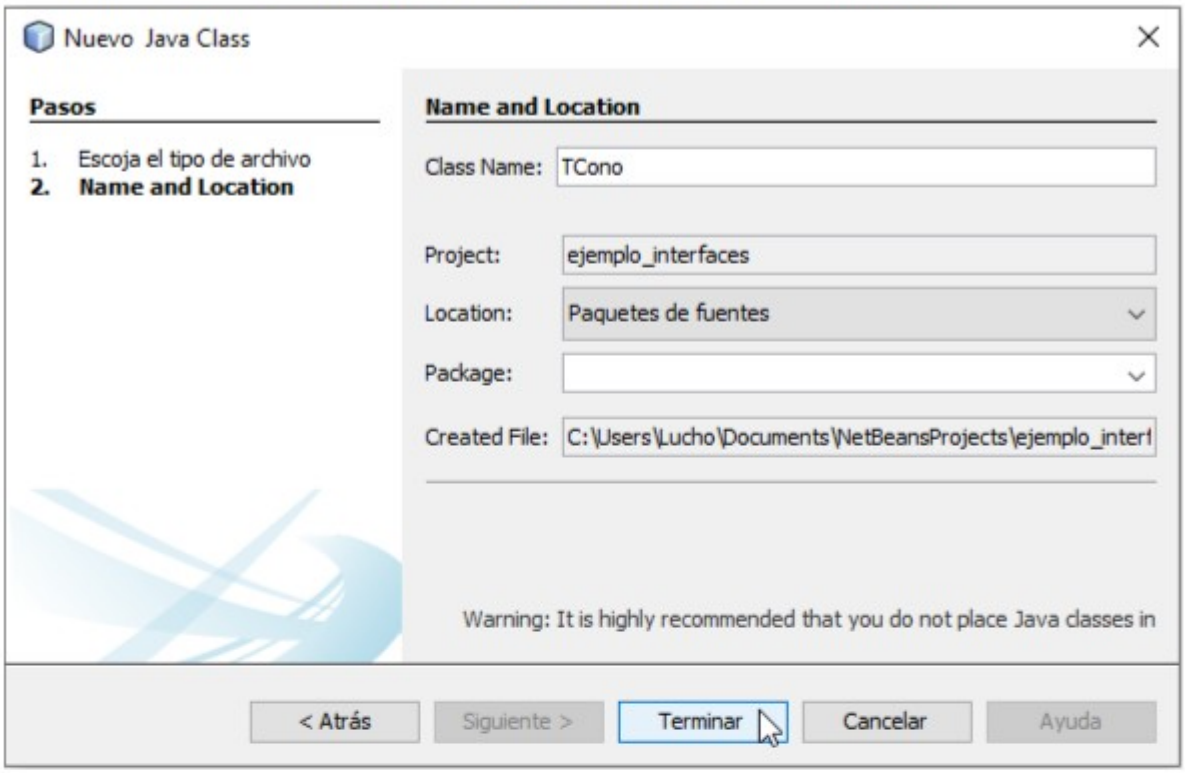
```
22
23  public float getRadio(){
24    return Radio;
25  }
26
27  public float getAltura(){
28    return Altura;
29  }
30
31  @Override
32  public double AreaTotal(){
33    return 2*Math.PI*(Altura+Radio);
34  }
35
36  @Override
37  public double Volumen(){
38    return Math.PI*Altura*Radio*Radio;
39  }
40
41 }
```

e) Creación clase *TCono*

- Para crear esta nueva clase en el proyecto, haga click derecho sobre el nodo ***paquetes de fuentes*** del proyecto ***ejemplo_interfaces***, o también sobre el mismo proyecto; después seleccione la opción ***Nuevo*** y después ***Java Class…***, tal como se muestra abajo:



- En esta ventana que se presenta entrada "***Class Name***", ponga ***TCono*** *como* nombre de la clase y después haga click en el botón ***Terminar***.



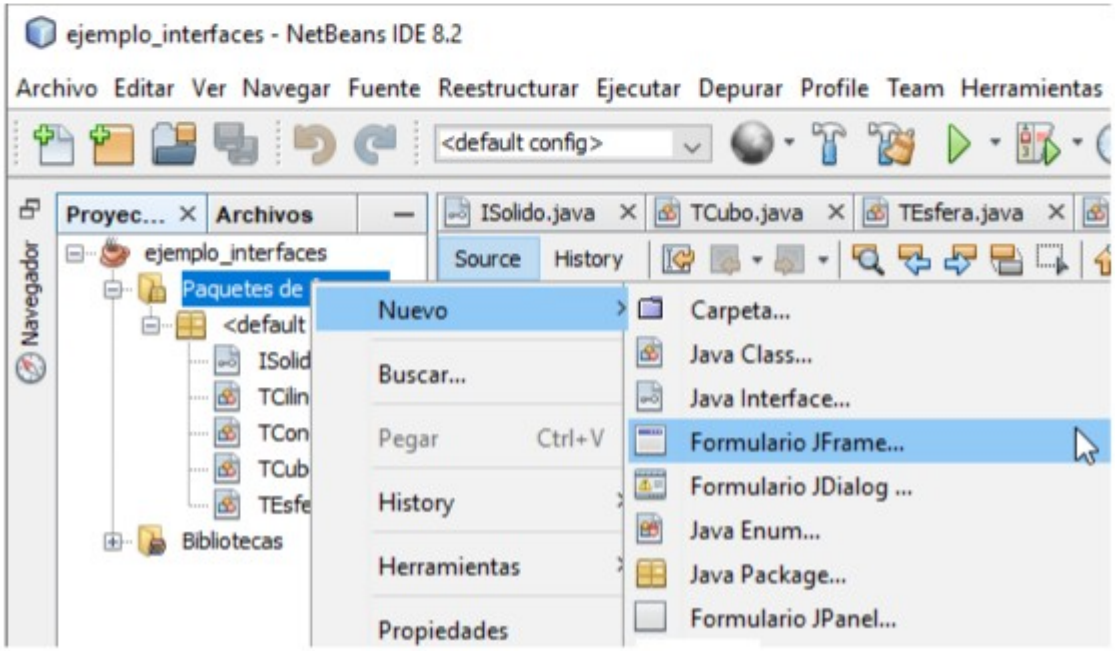
- Implemente la clase ***TCono*** (archivo *TCono*.java) como sigue:

```
1  public class TCono implements ISolido {
2
3    private float Radio;
4    private float Altura;
5
6    public TCono(){
7      Radio=0;
8      Altura=0;
9    }
10
11   public void setRadio(float Rad){
12     if(Rad>=0){
13       Radio=Rad;
14     }
15   }
16
17   public void setAltura(float Alt){
18     if(Alt>=0){
19       Altura=Alt;
20     }
21   }
```

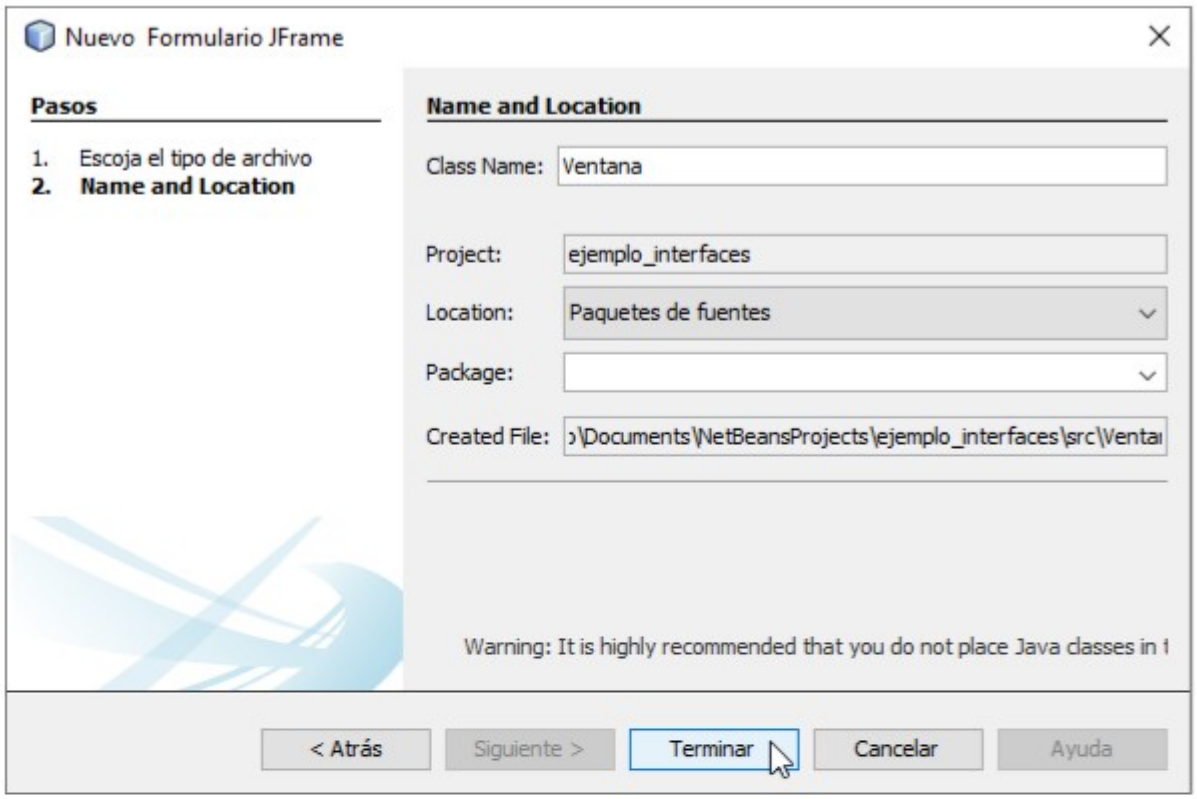
```
22
23  public float getRadio(){
24    return Radio;
25  }
26
27  public float getAltura(){
28    return Altura;
29  }
30
31  public double Generatriz(){
32    return Math.sqrt(Radio*Radio + Altura*Altura);
33  }
34
35  @Override
36  public double AreaTotal(){
37    return Math.PI*Radio*(Radio + Generatriz());
38  }
39
40  @Override
41  public double Volumen(){
42    return Math.PI*Altura*Radio*Radio/3;
43  }
44
45 }
```

5. Creación de la ventana para el proyecto

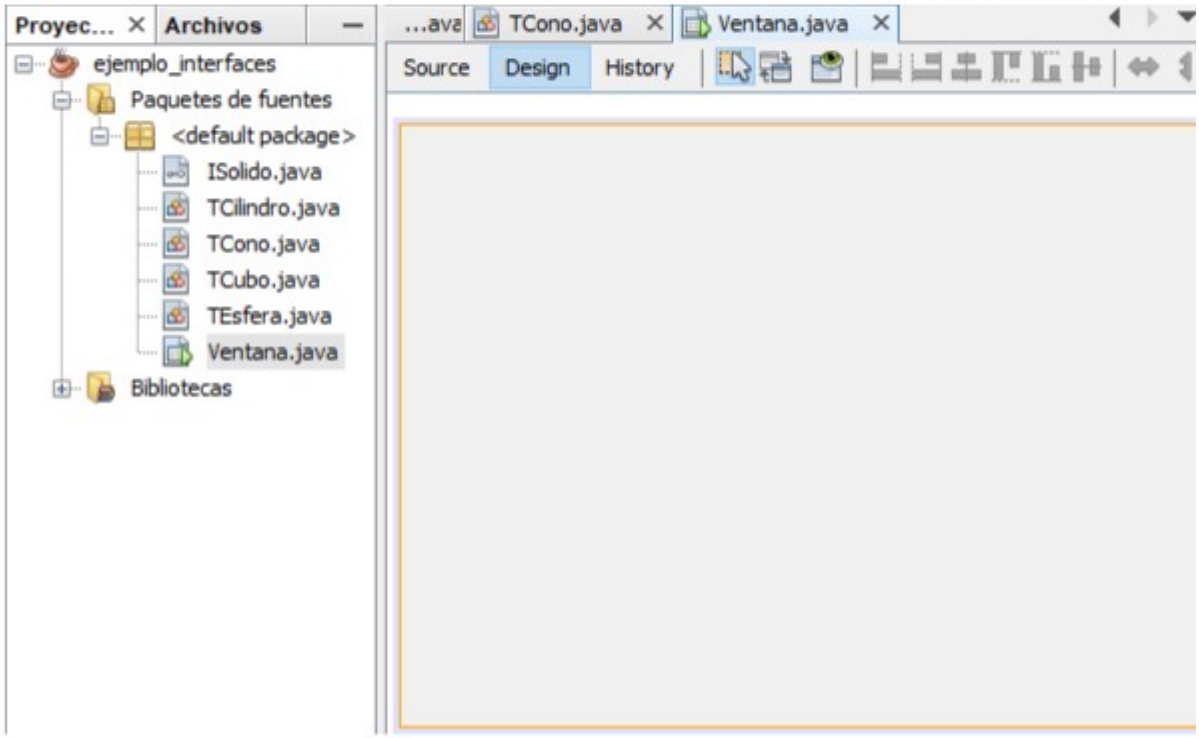
- Haga click derecho en nodo ***Paquetes de fuentes*** del proyecto, escoja la opción ***Nuevo*** y luego ***Formulario JFrame***, tal como se muestra en la siguiente imagen:



- En la ventana desplegada, en la entrada ***Class Name***, ingrese el nombre para la clase (***Ventana***) y haga click en el botón ***Terminar***.

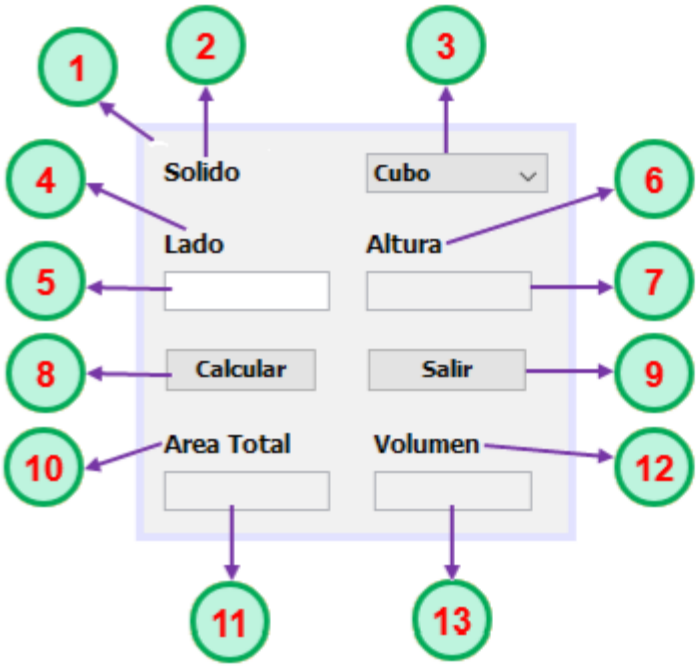


- Con ello la vista previa de la ventana en el IDE se vera de la siguiente manera:



6. Diseño gráfico de la ventana

- Asegúrese de diseñar la ventana de la siguiente manera, tomando en cuenta las indicaciones descritas en la tabla más abajo, considerando que los componentes apuntados por las flechas los encuentra en la ficha ***Menús Swing*** de la paleta de controles; por lo cual, es importante seguir el orden numérico indicado a la hora de arrastrar los componentes a la ventana.



7. Descripción de componentes de la ventana

Nº	Componente	Propiedad	Valor
1	*JFrame*	title	*Ejemplo interfaces*
		Generar centro	☑ (*Marcar la casilla*)
2	*JLabel*	text	Solido
3	*JComboBox*	*Nombre*	*cb1*
		model	*Click al botón de tres puntos, borre las opciones mostradas por defecto y escriba las siguientes:*
			Cubo ***Esfera*** ***Cilindro*** ***Cono***

Nº	Componente	Propiedad	Valor
4	*JLabel*	text	Lado
		Nombre	*lab1*
5	*JTextField*	*Nombre*	*tf1*
		text	*Borre el texto y déjelo en blanco*
6	*JLabel*	text	Altura
		Nombre	*lab2*
7	*JTextField*	*Nombre*	*tf2*
		text	*Borre el texto y déjelo en blanco*
8	*JButton*	*Nombre*	*B1*
		text	Calcular
9	*JButton*	*Nombre*	*B2*
		text	Salir
10	*JLabel*	text	Área Total
11	*JTextField*	*Nombre*	*tf3*
		text	*Borre el texto y déjelo en blanco*
		Editable	☐ (Desmarcar la casilla)
12	*JLabel*	text	Volumen
13	*JTextField*	*Nombre*	*tf4*
		text	*Borre el texto y déjelo en blanco*
		Editable	☐ (Desmarcar la casilla)

8. Definición de atributos e implementación de métodos de la ventana

- Asegúrese que el código de la clase para la ventana quede de la siguiente manera, teniendo en cuenta que el método ***initComponents*** no deben incluirlo, pues ya está implementando por *NetBeans* y no puede ser modificado:

```
1  public class Ventana extends javax.swing.JFrame {
2
3      public Ventana() {
4          initComponents();
5          lab2.setText("");
6      }
7
8      private TCubo CrearCubo(){
9          TCubo Cub;
10         Cub=new TCubo();
11         Cub.setLado(Float.parseFloat(tf1.getText()));
12         return Cub;
13     }
14
15     private TEsfera CrearEsfera(){
16         TEsfera Esf;
17         Esf=new TEsfera();
18         Esf.setRadio(Float.parseFloat(tf1.getText()));
19         return Esf;
20     }
21
22     private TCilindro CrearCilindro(){
23         TCilindro Cil;
24         Cil=new TCilindro();
25         Cil.setRadio(Float.parseFloat(tf1.getText()));
26         Cil.setAltura(Float.parseFloat(tf2.getText()));
27         return Cil;
28     }
29
30     private TCono CrearCono(){
31         TCono Con;
32         Con=new TCono();
33         Con.setRadio(Float.parseFloat(tf1.getText()));
34         Con.setAltura(Float.parseFloat(tf2.getText()));
35         return Con;
36     }
37
38     private void Mostrar(ISolido Sol){
39         tf3.setText(String.format("%.3f",Sol.AreaTotal()));
40         tf4.setText(String.format("%.3f",Sol.Volumen()));
41     }
42
```

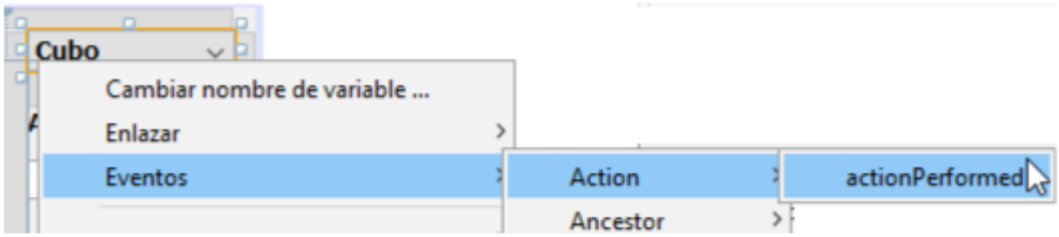
Luis Roberto Olascoaga Surmay

```
43  private void CambiarTitulo(){
44   lab1.setText("");
45   lab2.setText("");
46   switch(cb1.getSelectedIndex()){
47    case 0:lab1.setText("Lado");break;
48    case 1:lab1.setText("Radio");break;
49    case 2:case 3:lab1.setText("Radio");
50                  lab2.setText("Altura");
51                  break;
52   }
53   if(cb1.getSelectedIndex()<2){
54    lab2.setText("");
55    tf2.setText("");
56    tf2.setEnabled(false);
57   }
58   else{
59    tf2.setEnabled(true);
60   }
61  }
62
63  private void Calcular(){
64   ISolido Sol;
65   switch(cb1.getSelectedIndex()){
66    case 0:Sol=CrearCubo();break;
67    case 1:Sol=CrearEsfera();break;
68    case 2:Sol=CrearCilindro();break;
69    case 3:Sol=CrearCono();break;
70    default:Sol=null;
71   }
72   Mostrar(Sol);
73  }
74
75  //No inlcuir ni modificar esta parte desde aqui hacia abajo
76  @SuppressWarnings("unchecked")
77  // <editor-fold defaultstate="collapsed" desc="Generated Code">
78  private void initComponents() {
79     . . .
80     . . .
81  }// </editor-fold>
```

.
.
.

9. Implementación eventos opciones de menú de la ventana

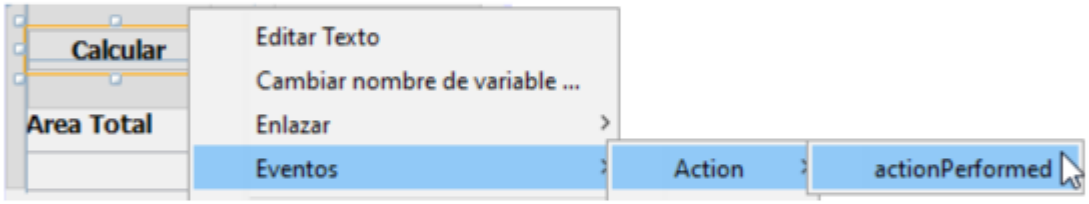
- Vaya al diseño de la ventana, haga click derecho sobre el *JComboBox* (***cb1***) y escoja las opciones ***Eventos*** + ***Action*** + ***actionPerformed***:



- El código para este evento es el siguiente:

```
private void cb1ActionPerformed(java.awt.event.ActionEvent evt) {
  CambiarTitulo();
}
```

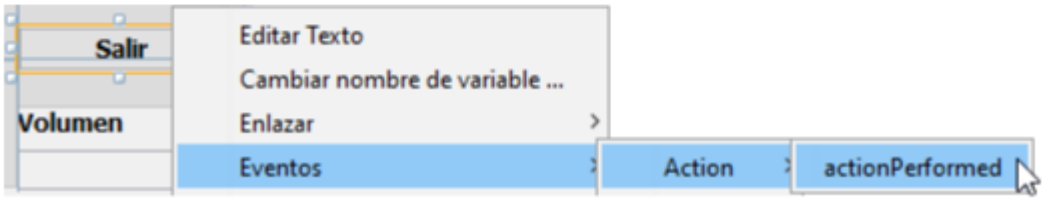
- Nuevamente en la vista de diseño de la ventana, haga click derecho sobre el botón ***Calcular*** y escoja las opciones ***Eventos*** + ***Action*** + ***actionPerformed***:



- El código para el evento de este botón es el siguiente:

```
private void B1ActionPerformed(java.awt.event.ActionEvent evt) {
  Calcular();
}
```

- Vaya al diseño de la ventana, haga click derecho sobre el botón ***Salir*** y escoja las opciones ***Eventos*** + ***Action*** + ***actionPerformed***:

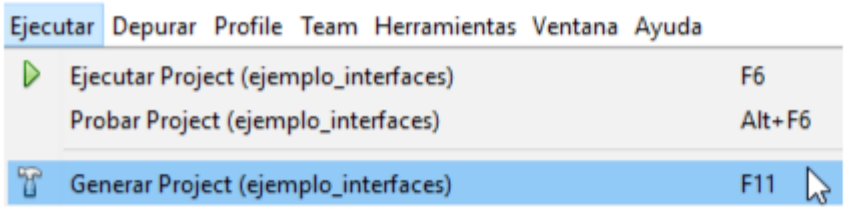


- El código para el evento de este botón es el siguiente:

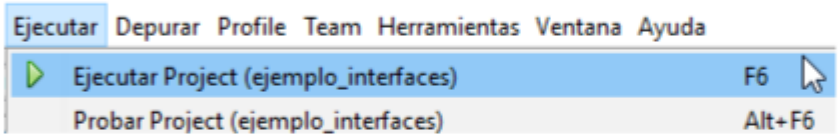
```
private void B2ActionPerformed(java.awt.event.ActionEvent evt) {
  dispose();
}
```

10. Compilación y ejecución del programa.

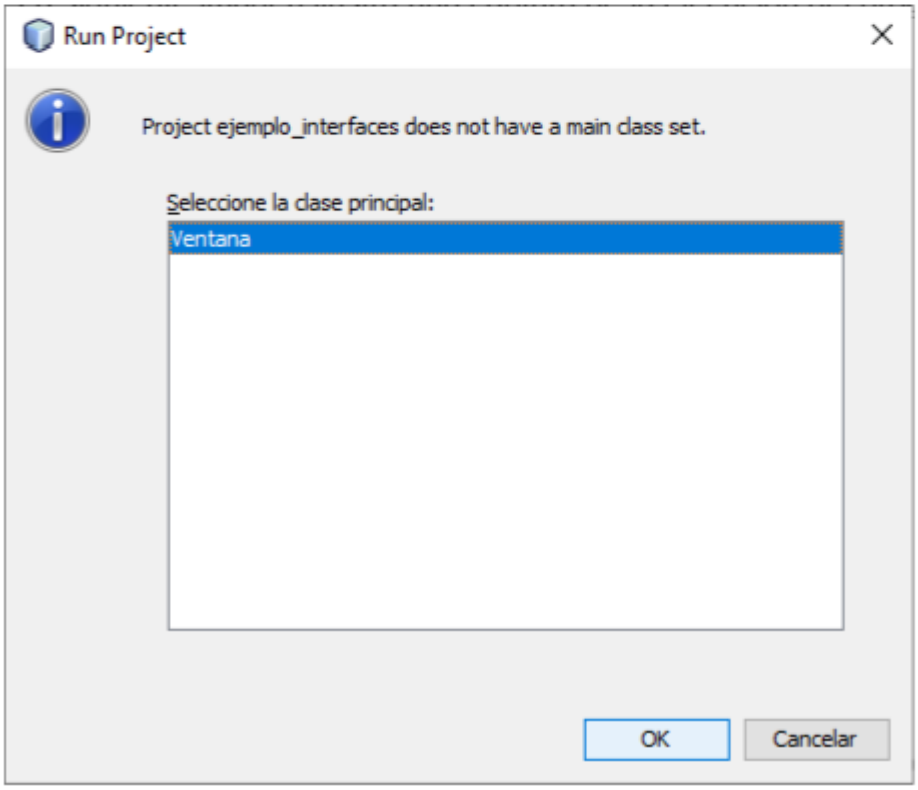
- Para compilar el programa vaya por la opción "***Ejecutar***" + "***Generar Project (ejemplo_interfaces)***" del menú principal o pulse la tecla F11.



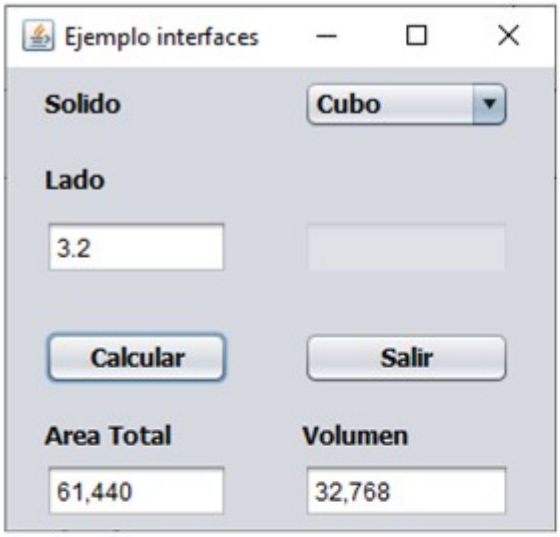
- Corrija los errores según los mensajes indicados por el compilador comparando con el código fuente anterior; luego puede correr el programa con la opción "***Ejecutar***" + "***Ejecutar Project (ejemplo_interfaces)***" del menú principal, o pulsando la tecla F6:



- Seguidamente, y solo cuando se ejecuta el programa por primera vez, *NetBeans* le mostrará el siguiente cuadro de dialogo para usar como clase principal la clase de la ventana (el ***JFrame***), de modo que haremos click en el botón ***OK***:



➲ Las siguientes son imágenes que ilustran capturas de la ejecución del programa:



Actividad propuesta

1. Una interfaz puede tener métodos abstractos si o no?, por que?.

2. Señale las diferencias existentes entre las relaciones de generalización (herencia) y realización.

3. Establezca y explique las similitudes existentes entre las relaciones de generalización y realización.

4. Es posible hacer herencia entre interfaces si o no?. por que?. Si la respuesta es afirmativa cual es la sintaxis seguida en Java para implementar herencia entre interfaces. Cite un ejemplo al respecto.

5. Una interfaz puede tener métodos abstractos si o no?, por que?.

6. En muchos lenguajes las interfaces se suelen usar como alternativa o soporte a la herencia múltiple. Explique por que la interfaz no se puede considerar formalmente como herencia múltiple, que limitaciones hay al respecto. Ilustre con un ejemplo sus argumentos.

7. Por qué decimos que una instancia de una interfaz tiene un papel o rol equivalente a una instancia de una clase abstracta padre?

8. Para que se utiliza el operador ***instanceof*** de Java.

9. Modifique aplicación anterior de modo que:

- No haga los cálculos cuando algún campo de entrada (valor de los ***JTextField***) este vacío; en cuyo caso muestre un mensaje de error adecuando, usando un ***JOptionPane***.
- No permita que se escriban valores no numéricos en los campos de entrada, excluyendo también valores negativos.

10. Añada en este proyecto clases para los siguientes poliedros regulares: tetraedro, octaedro, dodecaedro e icosaedro, de modo que las clases para estos solidos también implementen la interface ***ISolido*** y sean instanciadas (usadas) en la ventana.

Luis Roberto Olascoaga Surmay