

Eventos

Los listeners son interfaces Java incluidos en el paquete que se deben importar:

```
java.awt.event.*;
```

Los componentes que procesan los eventos deben implementar la interfaz *ActionListener*, que incluye el método *actionPerformed(ActionEvent e)*, el cual debe ser implementado.

Objetivo de aprendizaje

Al finalizar este ejercicio, tendrás la capacidad para definir oyentes o listeners para el tratamiento de eventos generados mediante la interacción de usuario en la interfaz gráfica.

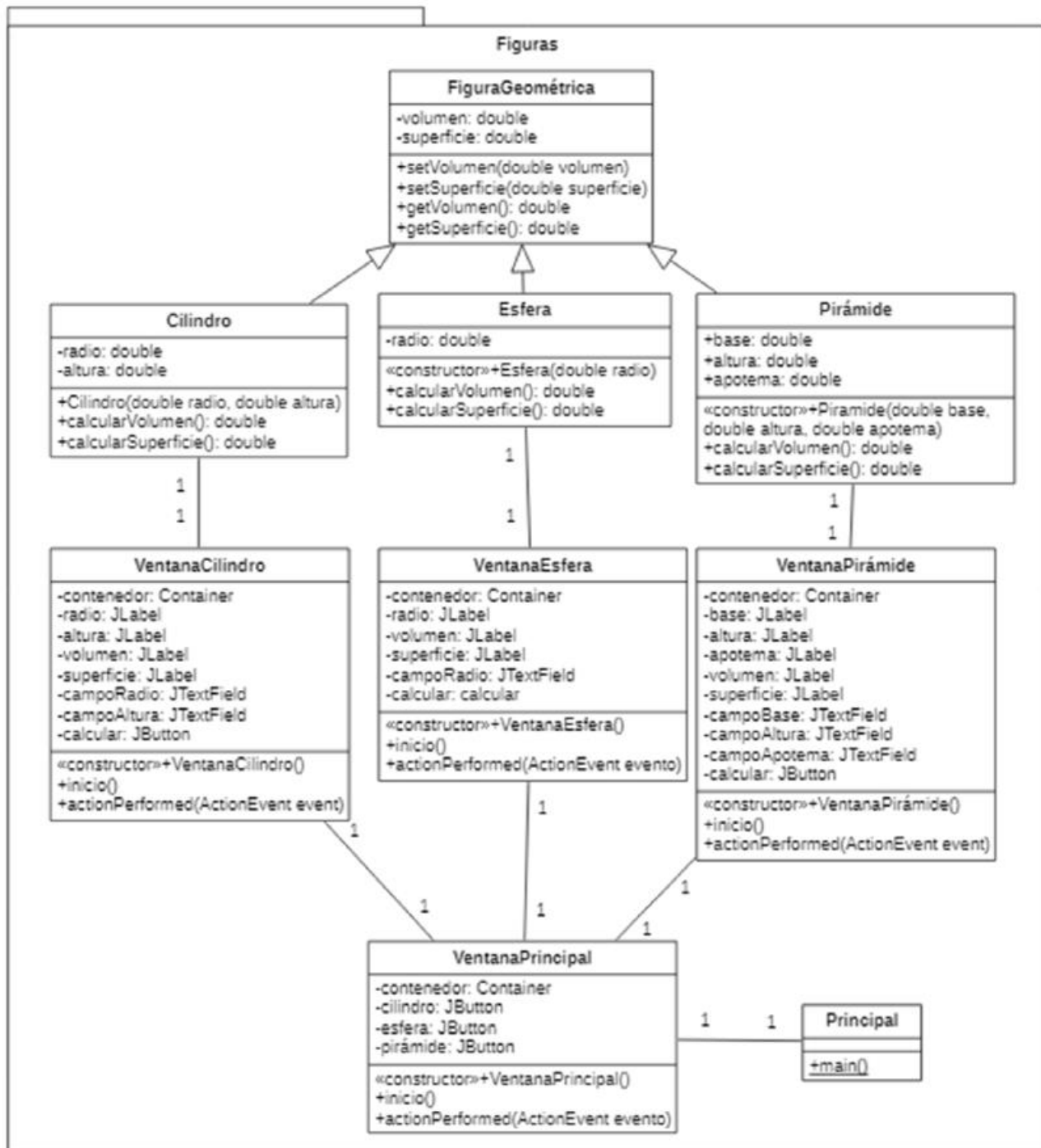
Enunciado: Figuras geometricas

Se requiere desarrollar un programa con interfaz gráfica de usuario que permita calcular el volumen y superficie de varias figuras geométricas. Las figuras geométricas son el cilindro, la esfera y la pirámide.

- Para el cilindro se solicitan su radio y altura (en centímetros).
- Para la esfera, su radio (en centímetros).
- Para la pirámide, su base, altura y apotema (en centímetros).

Una vez ingresados estos datos, el programa calcula el volumen y superficie de cada figura. Para desarrollar el programa se debe crear una jerarquía de clases para las diferentes figuras geométricas requeridas.

DIAGRAMA DE CLASES



Explicación del diagrama de clases

Se ha definido un paquete denominado "Figuras" que incluye un conjunto de clases. El punto de entrada al programa es la clase Principal que cuenta con el método main. Mediante una relación de asociación la clase Principal está relacionada con la clase VentanaPrincipal que posee atributos privados para identificar tres botones (JButton) que abrirán las ventanas respectivas de las figuras geométricas: cilindro, esfera y pirámide.

La clase VentanaPrincipal cuenta con un constructor y métodos para generar la ventana gráfica con sus componentes (inicio) y gestionar los diferentes eventos surgidos al interactuar con esta ventana (actionPerformed).

La clase VentanaPrincipal está conectada por medio de relaciones de asociación con las clases VentanaCilindro, VentanaEsfera y VentanaPirámide. Las multiplicidades de estas relaciones son una en cada extremo, lo que indica que se creará una sola ventana para cada tipo de figura geométrica.

Las clases VentanaCilindro, VentanaEsfera y VentanaPirámide cuentan con atributos para modelar los diferentes componentes gráficos de la ventana: un contenedor de componentes gráficos (Container), las etiquetas de los campos de texto y valores a calcular (JLabel), campos de texto para ingresar datos (JTextField) y un botón para calcular el volumen y superficie de las figuras (JButton).

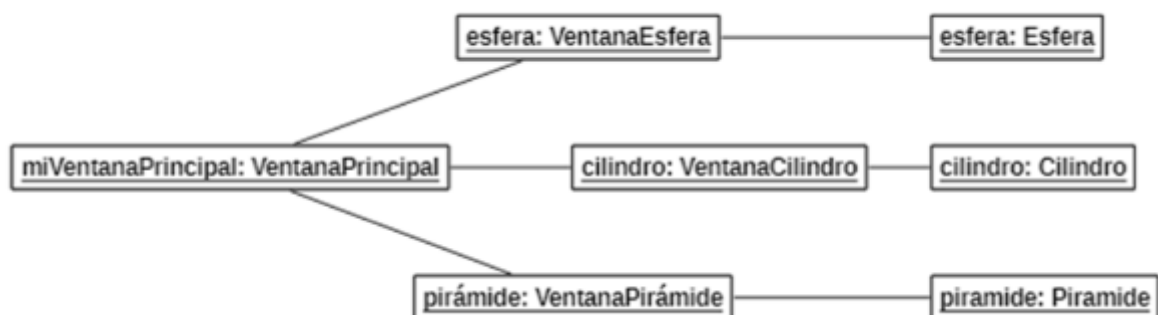
Así, en primer lugar, la clase VentanaCilindro se conecta con la clase Cilindro por medio de una relación de asociación que tiene una multiplicidad de uno en cada extremo. Por lo tanto, la VentanaCilindro creará un único objeto Cilindro. Esta clase Cilindro cuenta con dos atributos privados que identifican el radio y altura del cilindro, con el constructor respectivo que inicializa estos atributos y métodos para calcular la superficie y volumen del cilindro.

En segundo lugar, la clase VentanaEsfera se conecta con la clase Esfera utilizando una relación de asociación con multiplicidad de uno en cada extremo; la VentanaEsfera creará un único objeto Esfera y este estará asociado a una única VentanaEsfera. La clase Esfera cuenta con un único atributo privado para identificar el radio de la esfera, con un constructor que inicializa el atributo y métodos para calcular la superficie y volumen de la esfera.

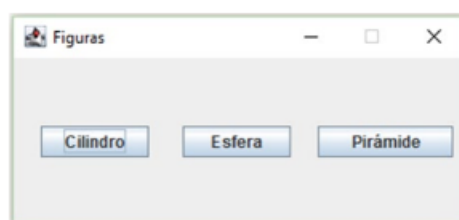
En tercer lugar, la clase VentanaPirámide se relaciona con la clase Pirámide por medio de una asociación con multiplicidad 1 en cada extremo. La clase Pirámide define tres atributos privados para identificar la base, altura y apotema de la pirámide y tiene con un constructor que inicializa sus atributos y métodos para calcular la superficie y volumen de la pirámide.

Finalmente, las clases Cilindro, Esfera y Pirámide son subclases de FiguraGeométrica, la cual es la clase padre que define atributos compartidos por las tres subclases: volumen y superficie, y posee métodos get y set para estos atributos.

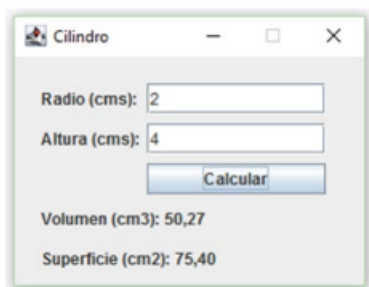
Diagrama de objetos



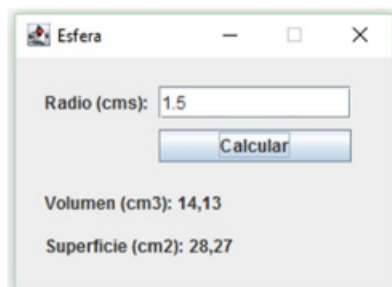
Ejecución del programa



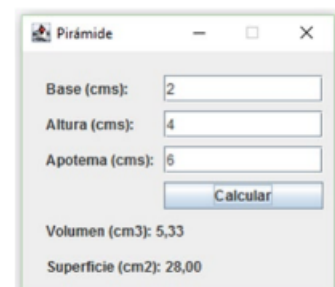
a) Ventana Principal



b) Cilindro



c) Esfera



d) Pirámide

Ejercicios propuestos

- Agregar el cálculo del volumen y superficie de otras figuras geométricas como el cubo y el prisma.
- Agregar a cada figura geométrica correspondiente su imagen correspondiente.

Instrucciones del ejercicio

Clase	Método	Descripción
Math	<i>double</i> PI	Valor <i>double</i> que representa la constante Pi.
	<i>double</i> pow(<i>double</i> a, <i>double</i> b)	Retorna el valor del primer argumento elevado a la potencia del segundo argumento.
Double	<i>double</i> parsedouble(<i>Strings</i>)	Devuelve un <i>double</i> inicializado con el valor representado por el <i>String</i> especificado.
String	<i>String</i> format(<i>String</i> formato, <i>Object</i> ... args)	Retorna un <i>String</i> formateado utilizando el formato y argumentos especificados.
JFrame	<i>JFrame</i> ()	Constructor de la clase <i>JFrame</i> .
	<i>void</i> setTitle(<i>String</i> título)	Establece el título de la ventana con el <i>String</i> especificado.
	<i>void</i> setSize(<i>int</i> x, <i>int</i> y)	Cambia el tamaño del componente para que tenga una anchura x y una altura y.
	<i>void</i> setLocationRelativeTo(<i>Component</i> c)	Establece la ubicación de la ventana en relación con el componente especificado.
	<i>void</i> setDefaultCloseOperation(<i>opciones</i>)	Usado para especificar una de las siguientes opciones del botón de cierre: EXIT_ON_CLOSE, HIDE_ON_CLOSE, DISPOSE_ON_CLOSE o DO_NOTHING_ON_CLOSE.
	<i>void</i> setResizable(<i>boolean</i> resizable)	Para evitar que se cambie el tamaño de la ventana.
	<i>void</i> setVisible(<i>boolean</i> b)	Muestra u oculta la ventana según el valor del parámetro b.
ActionListener	<i>void</i> actionPerformed(<i>actionEvent</i> e)	Se invoca cuando ocurre un evento.
Container	<i>Container</i> getContentPane()	Retorna el objeto <i>ContentPane</i> de la ventana.
	<i>void</i> setLayout(<i>LayoutManager</i> mgr)	Establece el <i>layout</i> de la ventana.
	<i>Component</i> add(<i>Component</i> comp)	Añade el componente especificado al final del contenedor.
Component	<i>void</i> addActionListener(<i>this</i>)	Añade un oyente de eventos al componente actual.
	<i>void</i> setBounds(<i>int</i> x, <i>int</i> y, <i>int</i> ancho, <i>int</i> alto)	Mueve y cambia el tamaño del componente.
JLabel	<i>JLabel</i> ()	Constructor de la clase <i>JLabel</i> .
	<i>void</i> setText(<i>String</i> text)	Define una línea de texto que mostrará este componente.

<i>JTextField</i>	<i>JTextField()</i>	Constructor de la clase <i>JTextField</i> .
	<i>String getText()</i>	Retorna el texto contenido en el componente de texto.
<i>JButton</i>	<i>JButton()</i>	Constructor de la clase <i>JButton</i> .
	<i>void setText(String text)</i>	Define una línea de texto que mostrará este componente.
	<i>int getSelectedIndex()</i>	Devuelve el índice seleccionado.
<i>Event</i>	<i>Object getSource()</i>	El objeto sobre el cual el evento inicialmente ha ocurrido.
<i>JOptionPane</i>	<i>void showMessageDialog</i> <i>(Component componentePadre,</i> <i>Object mensaje)</i>	Crea un cuadro de diálogo.