Objetivos de aprendizaje

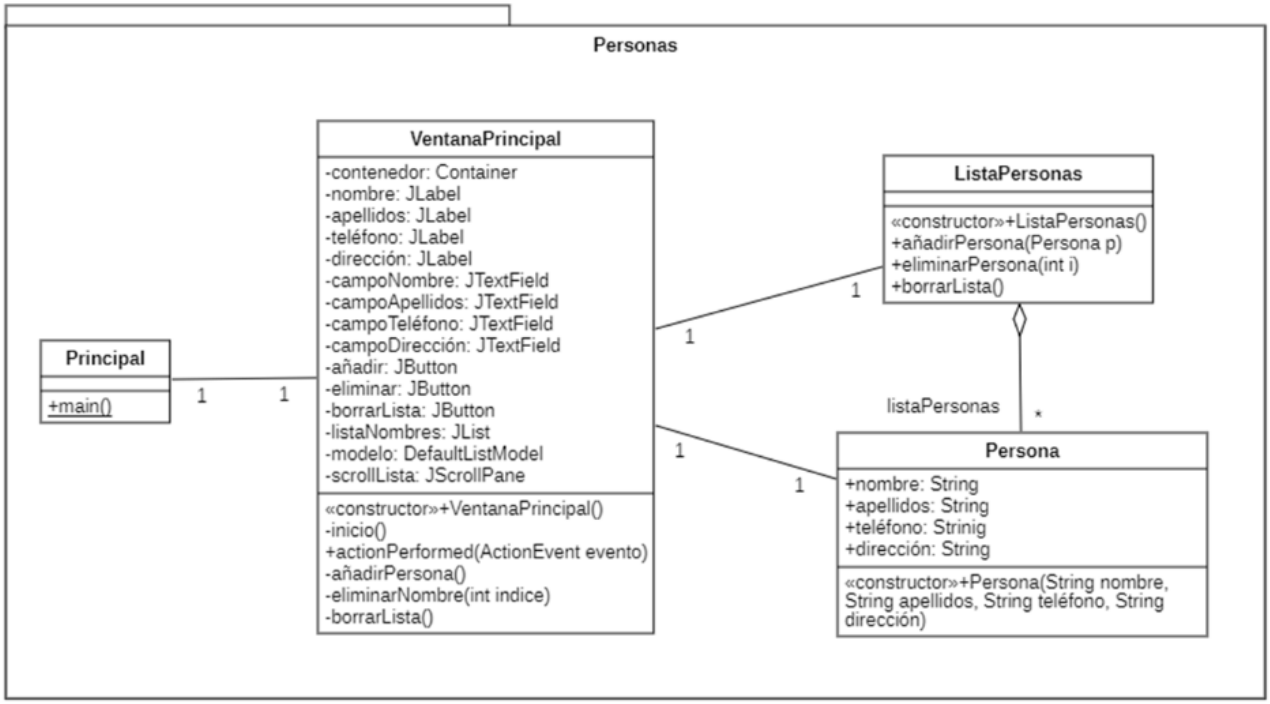
Al finalizar este ejercicio, el lector tendrá la capacidad para:

- Desarrollar interfaces de gráficas de usuario básicas.
- Conocer y aplicar diferentes componentes básicos de Java.

Enunciado: Persona

Se requiere desarrollar un programa con una interfaz gráfica de usuario que genere una ventana para solicitar los datos de una persona: nombre, apellidos, dirección y teléfono, todos de tipo String. Una vez se reciban estos datos, la persona se debe agregar a una lista de personas.
La lista se puede consultar y si se selecciona una persona específica, es posible eliminar la persona seleccionada. También se permite borrar todas las personas de la lista.

Diagrama de clases



Explicación del diagrama de clases

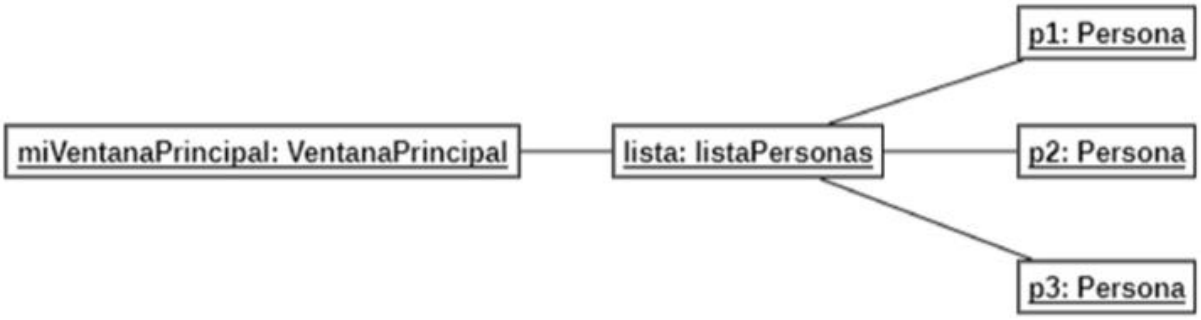
Se ha definido un paquete denominado “Personas” que incluye un conjunto de clases. El punto de entrada al programa es la clase Principal que cuenta con el método main. La clase Principal está vinculada mediante una relación de asociación con la clase VentanaPrincipal que posee atributos privados para identificar los diferentes componentes gráficos de la ventana: un contenedor de componentes gráficos (Container), etiquetas (JLabel), campos de texto (JTextField), botones (JButton), una lista (JList y DefaultListModel) y una barra de desplazamiento vertical (JScrollPane). La clase VentanaPrincipal cuenta con un constructor y métodos para generar la ventana gráfica con sus componentes (inicio) y para gestionar los diferentes eventos surgidos al interactuar con esta ventana (actionPerformed).

La clase VentanaPrincipal se relaciona con las clases ListaPersonas y Persona a la par que se generan los diferentes eventos, cuando el usuario interactúa con los botones presentes en la ventana. Estas relaciones de asociación tienen una multiplicidad de 1 a 1, ya que la VentanaPrincipal tiene una única lista de personas y cuando crea una persona, solo es posible crear una a la vez.

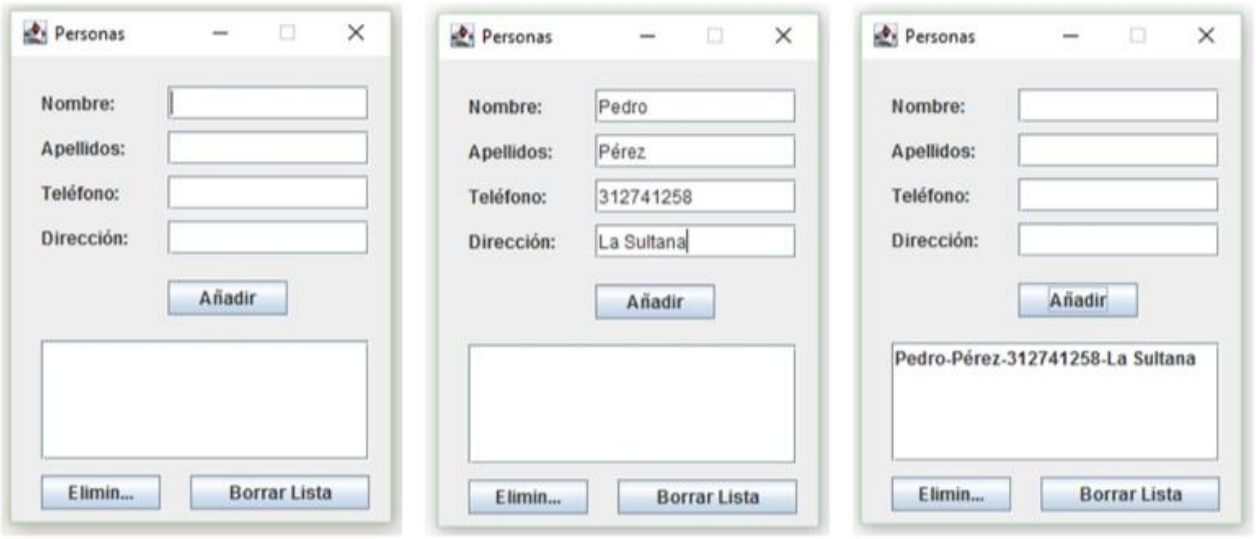
La clase ListaPersonas tiene una relación de agregación con la clase Persona. La relación de agregación se expresa en UML por medio de una línea continua que tiene un rombo claro adyacente a la clase que representa el todo. En este caso, la clase que representa el todo es ListaPersonas, la cual está constituida de objetos tipo Persona.

Las clases ListaPersona y Persona tienen sus atributos, constructor y métodos correspondientes. Los atributos de Persona son: nombre, apellidos, dirección y teléfono y tiene un constructor para inicializar estos atributos. La clase ListaPersona tiene un constructor y métodos para añadir, eliminar personas y borrar la lista de personas.

Diagrama de objetos



Ejecución del programa



a) Ventana Principal b) Ingreso de persona c) Lista de personas

Ejercicios propuestos

- Agregar la funcionalidad para editar los datos de una persona seleccionada de la lista de personas.
- Agregar la funcionalidad para generar mensajes de alerta cuando se ingrese una persona cuyos nombres y apellidos ya se encuentren en la lista.

Instrucciones Java del ejercicio

Clase	Método	Descripción
JFrame	*JFrame()*	Constructor de la clase JFrame.
	void setTitle(String título)	Establece el título de la ventana con el *String* especificado.
	void setSize(int x, int y)	Cambia el tamaño del componente para que tenga una anchura *x* y una altura *y*.
	void setLocationRelativeTo (Component c)	Establece la ubicación de la ventana en relación con el componente especificado.
	void setDefaultCloseOperation (opciones)	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.
	void setResizable(boolean resizable)	Para evitar que se cambie el tamaño de la ventana.
	void setVisible(boolean b)	Muestra u oculta la ventana según el valor del parámetro b.
ActionListener	*ActionListener*	Interface que debe ser implementada para gestionar eventos.
	void actionPerformed(actionEvent e)	Se invoca cuando ocurre un evento.
Container	*Container getContentPane()*	Retorna un objeto *ContentPane* de la ventana.
	void setLayout(LayoutManager mgr)	Establece el *layout* de la ventana.
	Component add(Component comp)	Añade el componente especificado al final del contenedor.
Component	*void addActionListener(this)*	Añade un oyente de eventos al componente actual.
	void setBounds(int x, int y, int ancho, int alto)	Mueve y cambia el tamaño del componente.
JLabel	*JLabel()*	Constructor de la clase *JLabel*.
	void setText(String text)	Define una línea de texto que mostrará este componente.
JTextField	*JTextField()*	Constructor de la clase *JTextField*.
	String getText()	Retorna el texto contenido en el componente de texto.

Clase	Método	Descripción
JButton	*JButton()*	Constructor de la clase *JButton*.
	void setText(String text)	Define una línea de texto que mostrará este componente.
JList	*JList()*	Constructor de la clase JList.
	void setText(String text)	Define una línea de texto que mostrará este componente.
	void setSelectionMode(int modoSelección)	Establece el modo de selección de la lista.
	void setModel(ListModel<E> model)	Establece el modelo que representa el contenido de la lista.
	int getSelectedIndex()	Devuelve el índice seleccionado.
DefaultList Model	*DefaultListModel()*	Constructor de la clase *DefaultListModel*.
	void addElement(Element e)	Añade el componente especificado al final de la lista.
	void removeElementAt(int índice)	Elimina el componente en el índice especificado.
	void removeAllElements()	Elimina todos los componentes de la lista y coloca su tamaño en cero.
	void clear()	Elimina todos los elementos de la lista.
JScrollPane	*JScrollPane()*	Constructor de la clase *JScrollPane*.
	void setViewportView(Component view)	Crea una ventana gráfica si es necesario y luego establece su vista.
Event	*Object getSource()*	El objeto sobre el cual el evento inicialmente ha ocurrido.
JOptionPane	*void showMessageDialog (Component componentePadre, Object mensaje)*	Crea un cuadro de diálogo.