

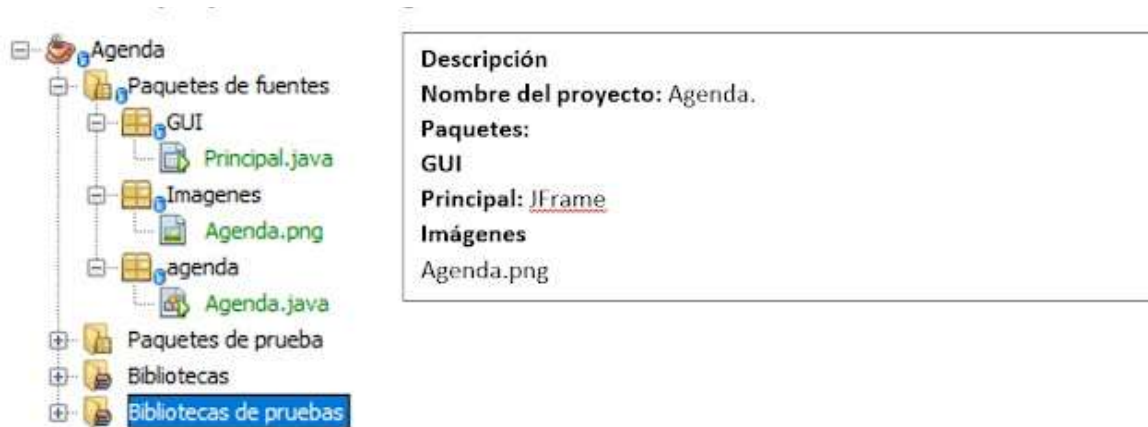
CREAR UNA AGENDA TELEFÓNICA EN JAVA NETBEANS USANDO ARREGLOS

Siguiendo el presente tutorial podrás crear una **agenda telefónica en Java**, utilizando el IDE NetBeans, además manejando **Arreglos o Arrays en Java** para almacenar los datos, para este proyecto utilizaremos una interfaz gráfica (GUI); por medio de **JFrame**, al finalizar este tutorial afianzarás tus conocimientos acerca de arreglos, manejo de JTable, campos de texto (*TextField*) y demás controles de formularios.

Crear una Agenda telefónica en Java NetBeans usando Arreglos

Sigue los siguientes pasos:

1. Crea un proyecto con la siguiente estructura:



2. Diseña el JFrame de la siguiente manera:



3. Coloca los nombres de los controles

Asigna los siguientes nombres a los controles del formulario.

Control	Tipo
<i>txtIdentificacion</i>	TextField
<i>txtNombres</i>	TextField
<i>txtApellidos</i>	TextField
<i>txtTelefono</i>	TextField
<i>txtDireccion</i>	TextField
<i>btnGuardar</i>	Button
<i>btnLimpiar</i>	Button
<i>btnVerAgenda</i>	Button
<i>jTContactos</i>	JTable

4. Eliminando el modelo que trae por defecto el JTable.

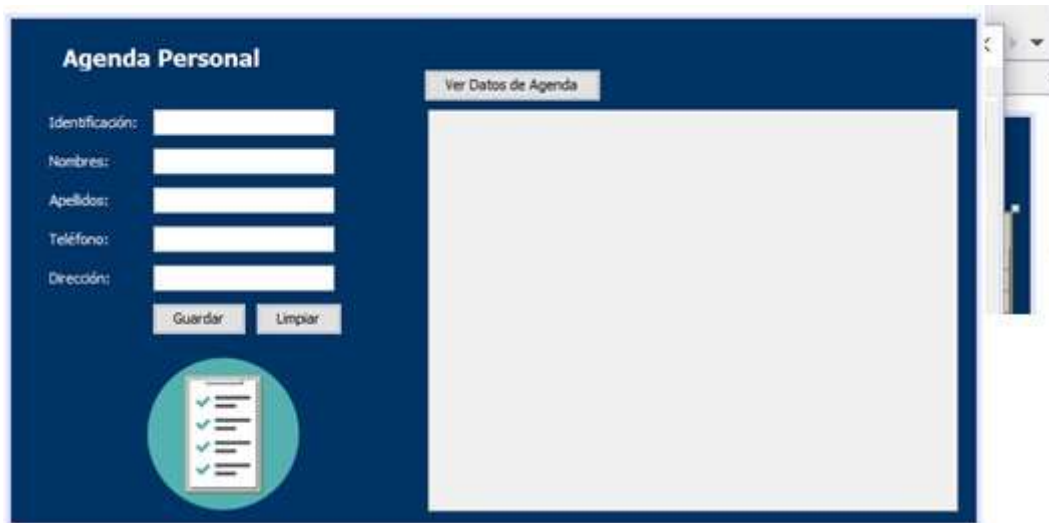
Clic derecho sobre la tabla y selecciona **Contenido de la tabla...**



Dirígete a la pestaña **Columnas**, selecciona una por una y presiona **Suprimir**.



En modo diseño la tabla debe quedar vacía, así.



5. Creando la clase «Contactos».

Ahora debemos crear una clase que nos permitirá crear “Objetos” de tipo contacto que guardaremos en la agenda que será un “**ArrayList**”.

Crea un **paquete** llamado **Clases** y dentro crea una clase llamada **Contactos**.



La clase Contactos contará con los atributos o características que tendrá cada contacto de la agenda, que básicamente son los datos que están en el JFrame.

En la clase (*cada atributo*) debe contar con 2 métodos importantes: **get** y **set**.

Método **get**.

Permite obtener el valor correspondiente a un atributo de un objeto, por consiguiente si hacemos el llamado al método **getNombre()**, estaríamos obteniendo el nombre del contacto.

Método **set**.

Es útil para asignar un valor a un objeto tipo “Contacto”, ejemplo, si utilizamos **setNombre(“Juan”)**, estamos asignando al contacto el nombre Juan.

Código de la clase Contactos

```
package Clases;
public class Contactos {
    int identificacion;
    String nombres;
    String apellidos;
    String telefono;
    String direccion;
    public int getIdentificacion() {
        return identificacion;
    }
    public void setIdentificacion(int identificacion) {
        this.identificacion = identificacion;
    }
    public String getNombres() {
        return nombres;
    }
    public void setNombres(String nombres) {
        this.nombres = nombres;
    }
    public String getApellidos() {
        return apellidos;
    }
    public void setApellidos(String apellidos) {
        this.apellidos = apellidos;
    }
    public String getTelefono() {
        return telefono;
    }
    public void setTelefono(String telefono) {
        this.telefono = telefono;
    }
    public String getDireccion() {
        return direccion;
    }
    public void setDireccion(String direccion) {
        this.direccion = direccion;
    }
}
```

6. Guardando los datos en la Agenda

Debemos tener un lugar donde guardar los contactos, para este ejemplo los guardaremos en un ArrayList; dirígete al código del **JFrame** (Principal) y en la parte superior define el ArrayList que utilizaremos.

```
package GUI;

import java.util.ArrayList;

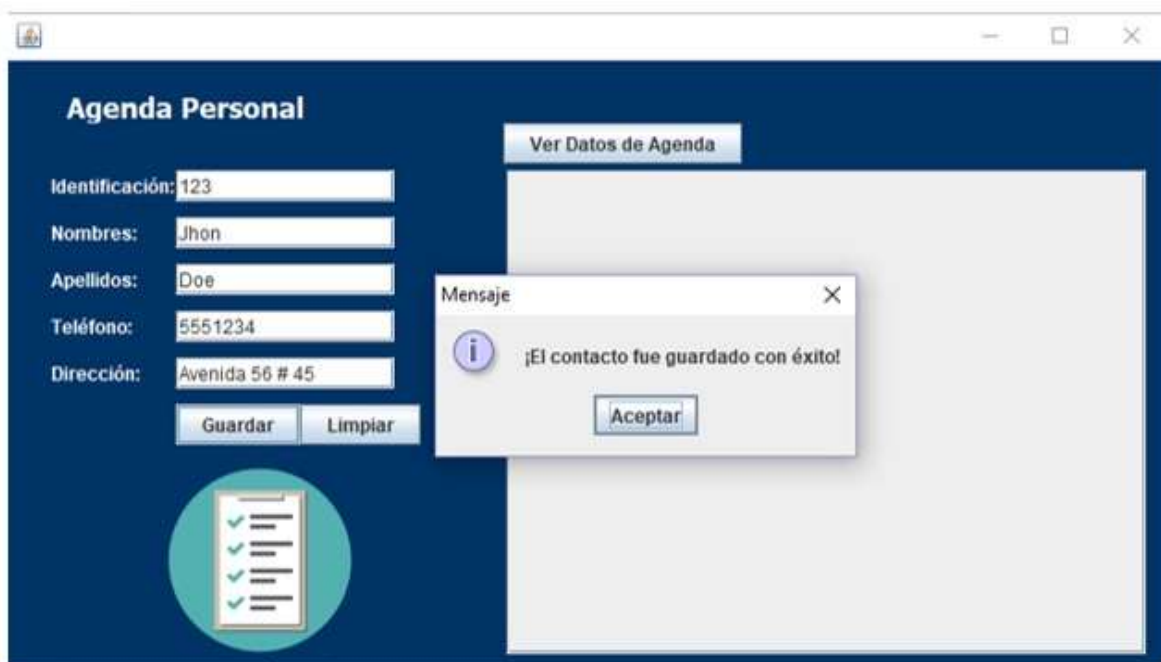
public class Principal extends javax.swing.JFrame {
    ArrayList Agenda = new ArrayList();
}
```

7. Código del botón Guardar.

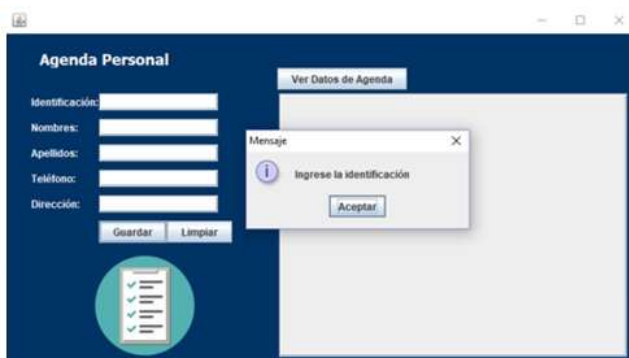
Debemos tener presente que en este botón lo primero que debemos hacer es validar que no queden campos vacíos, para posteriormente guardar los datos así.

```
private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
    String id = txtIdentificacion.getText();
    String nom = txtNombres.getText();
    String ape = txtApellidos.getText();
    String tel = txtTelefono.getText();
    String dir = txtDireccion.getText();
    if (!id.isEmpty()) {
        if (!nom.isEmpty()) {
            if (!ape.isEmpty()) {
                if (!tel.isEmpty()) {
                    if (!dir.isEmpty()) {
                        Contactos nuevo_contacto = new Contactos();
                        nuevo_contacto.setIdentificacion(Integer.parseInt(id));
                        nuevo_contacto.setNombres(nom);
                        nuevo_contacto.setApellidos(ape);
                        nuevo_contacto.setTelefono(tel);
                        nuevo_contacto.setDireccion(dir);
                        Agenda.add(nuevo_contacto);
                        JOptionPane.showMessageDialog(null, "El contacto fue guardado con éxito!");
                    } else {
                        JOptionPane.showMessageDialog(null, "Ingrese la dirección");
                    }
                } else {
                    JOptionPane.showMessageDialog(null, "Ingrese el teléfono");
                }
            } else {
                JOptionPane.showMessageDialog(null, "Ingrese el Apellido");
            }
        } else {
            JOptionPane.showMessageDialog(null, "Ingrese el Nombre");
        }
    } else {
        JOptionPane.showMessageDialog(null, "Ingrese la identificación");
    }
}
```

Al ejecutar el programa debe guardar los datos ingresados en el ArrayList, "Aunque todavía no podamos verlos".



Así mismo debe validar que ningún dato quede vacío.



Si en este punto el programa no logra estas funcionalidades, **“Guardar y Validar”**, debe revisar el código antes de seguir y corregir los errores; se recomienda no seguir hasta terminar esta parte.

8. Código del botón Limpiar

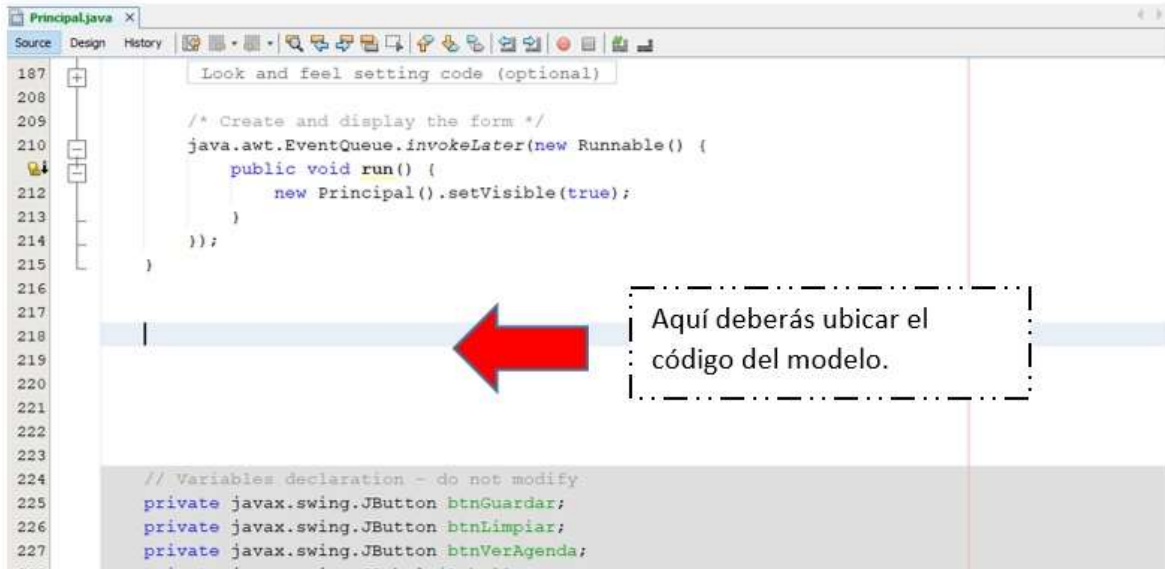
El botón limpiar se encarga de vaciar los campos para que el usuario empiece a ingresar los datos, ubicándolo en la primera casilla (Identificación).

```
private void btnLimpiarActionPerformed(java.awt.event.ActionEvent evt) {  
    txtIdentificacion.setText("");  
    txtNombres.setText("");  
    txtApellidos.setText("");  
    txtTelefono.setText("");  
    txtDireccion.setText("");  
    txtIdentificacion.requestFocusInWindow();  
}
```

9. Leer los datos de la Agenda (ArrayList) y mostrarlos en una Tabla (JTable).

Creando un modelo

El modelo de un jTable es la estructura (Filas y Columnas), para crear el modelo dirígete al código del formulario; el código del modelo deberás escribirlo fuera de todas las llaves del código, un ejemplo sería.



Código del modelo



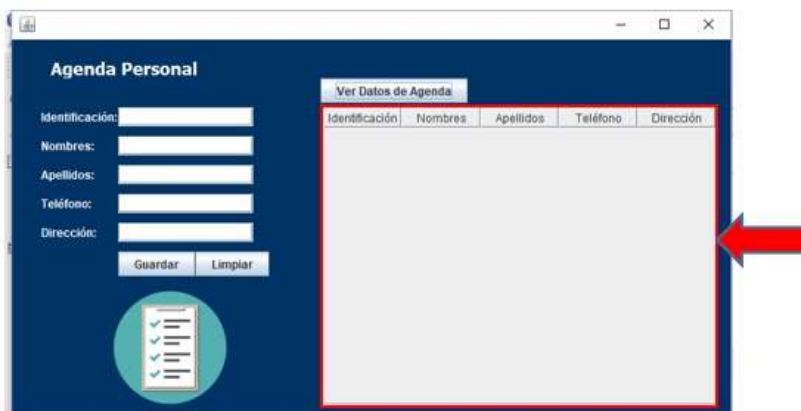
Como puedes observar en el código anterior detallamos las columnas que deseamos mostrar.

10. Mostrar el modelo.

Para mostrar el modelo debemos hacer el llamado del método `Crear_modelo()`, en el constructor del JFrame (Principal), dirígete al código del formulario, y hacemos el llamado así.

```
18 public Principal() {  
19     initComponents();  
20     Crear_Modelo();  
21 }  
22
```

Ahora ejecuta el proyecto y verás el modelo del JTable funcionando.



11. Mostrando la información de la Agenda en el JTable.

Debemos recordar que la información de la agenda está almacenada en un ArrayList, por lo tanto debemos recorrerlo para llenar el JTable (modelo).

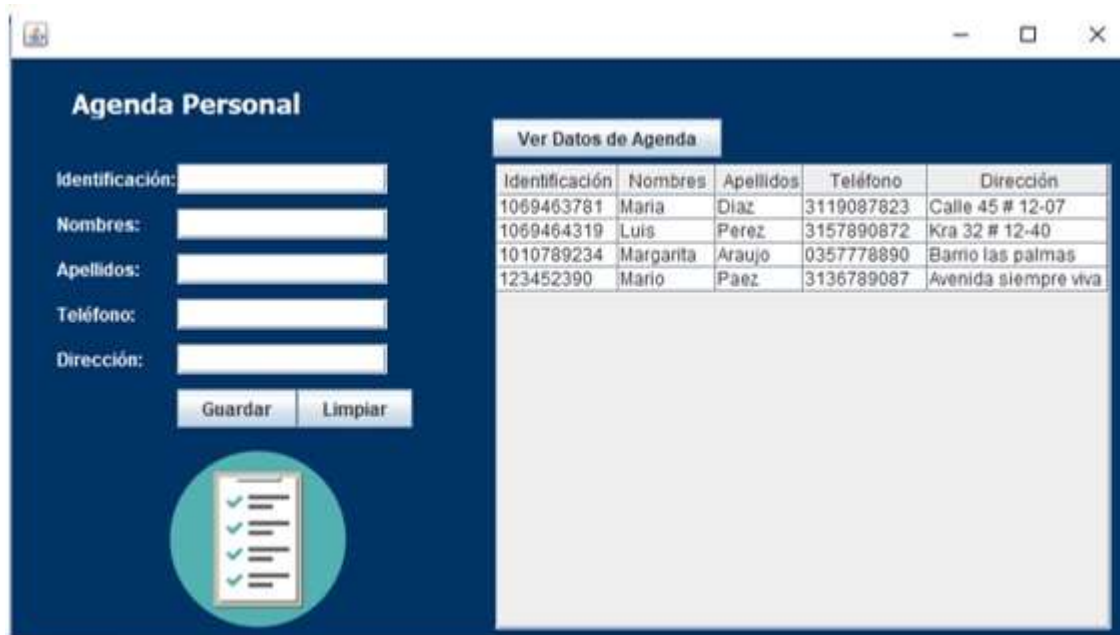
Código del botón Ver Datos de Agenda

```
private void btnVerAgendaActionPerformed(java.awt.event.ActionEvent evt) {  
    Object O[]=null;  
    for (int i = 0; i < Agenda.size(); i++) {  
        Contactos con = (Contactos) Agenda.get(i);  
        modelo.addRow(O);  
        modelo.setValueAt(con.getIdentificacion(), i, 0);  
        modelo.setValueAt(con.getNombres(), i, 1);  
        modelo.setValueAt(con.getApellidos(), i, 2);  
        modelo.setValueAt(con.getTelefono(), i, 3);  
        modelo.setValueAt(con.getDireccion(), i, 4);  
    }  
}
```

En el código anterior podemos observar cómo iteramos (Recorremos), el ArrayList y vamos obteniendo los datos para llenar cada fila.

12. Prueba definitiva del proyecto

Guarda varios registros y posteriormente presiona el botón Ver Datos de Agenda y deben cargarse en el listado de la tabla, así.



Identificación	Nombres	Apellidos	Teléfono	Dirección
1069463781	Maria	Diaz	3119087823	Calle 45 # 12-07
1069464319	Luis	Perez	3157890872	Kra 32 # 12-40
1010789234	Margarita	Araujo	0357778890	Barrio las palmas
123452390	Mario	Paez	3136789087	Avenida siempre viva

Añade las siguientes funciones al proyecto



Si llegaste a este punto pudiste crear una agenda telefónica en Java, por lo cual finalmente te invito a incorporar las siguientes funciones al proyecto:

- El botón guardar no debe permitir añadir una persona a la agenda con una identificación que ya este registrada.
- Al guardad los datos deberá Limpiar todos los campos y ubicar el cursor en el campo identificación.
- En botón “Ver datos de agenda”, se deberá indicar con un mensaje si la agenda está vacía.