



# Programación Orientada a Objetos

CRUD con persistencia en una base de  
datos



Sesión 05

Ing. David Mamani Pari



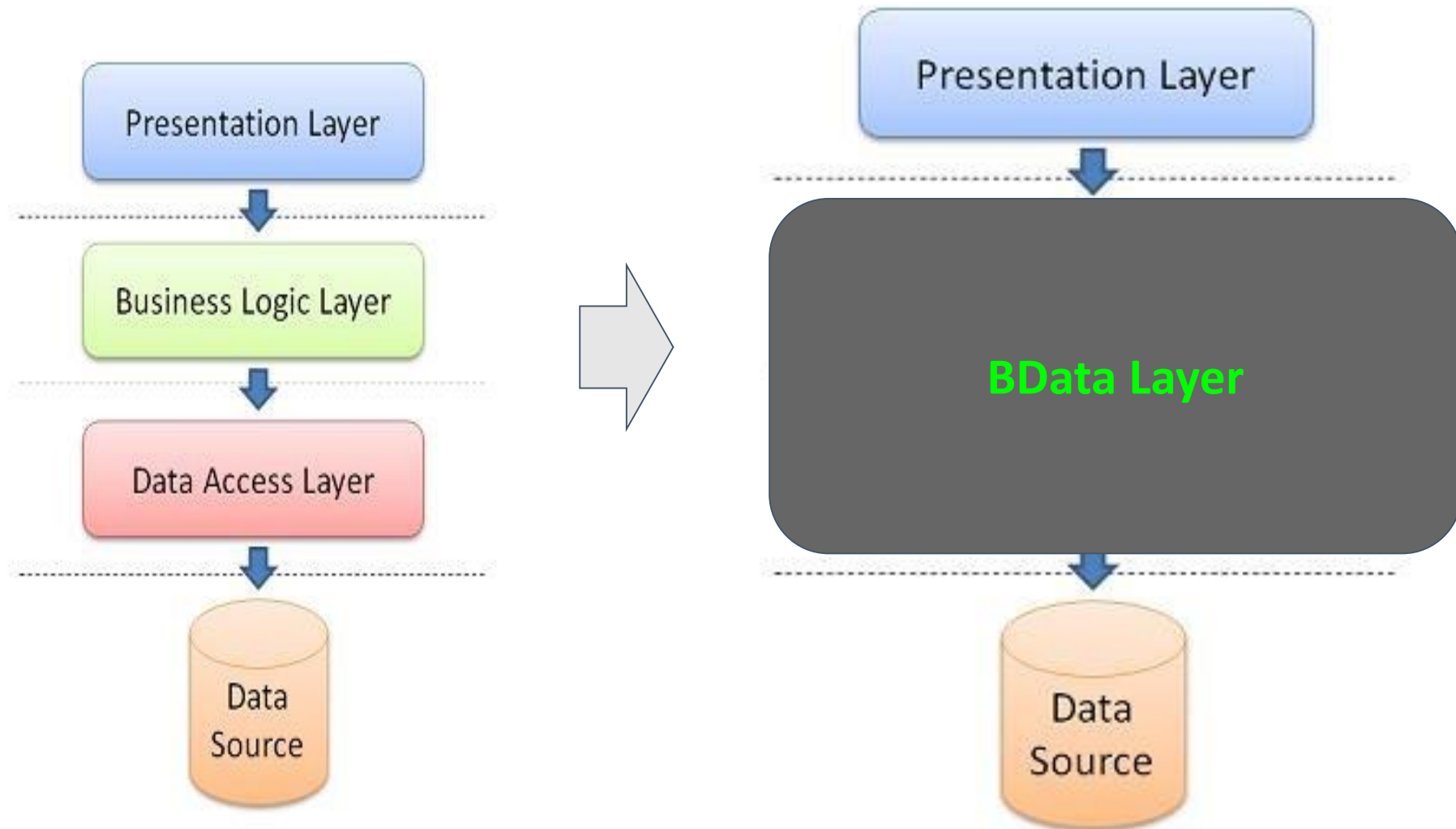
## Misión

“Somos una comunidad universitaria de la Iglesia Adventista del Séptimo Día que **modela** personas a fin de que sean Ingenieros de Sistemas **íntegros, misioneros, e innovadores** basados en la cosmovisión bíblica-cristiana para servir a Dios y a la humanidad”

## Visión



“Ser referentes en el mundo por el **modelamiento** de Ingenieros de Sistemas **íntegros, misionero e innovadores** con un estilo de vida saludable”



SystemMain@DMP

Archivo Ayuda

prueba1 Prueba 2 Preueba 3 Preueba 4 Preueba 5

## GESTIÓN DE CLIENTES

Dato a Buscar

Exportar

DNI/RUC:

Nombres:

Tipo:  ▼

#	Dni/Ruc	Nombres	Tipo
1	43631917	David Mamani Pari	Natural
2	43631918	Elias	General

Categoria		Marca	
IdCate	Nombre	IdMarca	Nombre
C001	Laptops	M001	Intel
C002	Audifonos	M002	AMD
C003	Tarjetas Video	M003	Lenovo
		M004	HP
		M005	NVIDIA

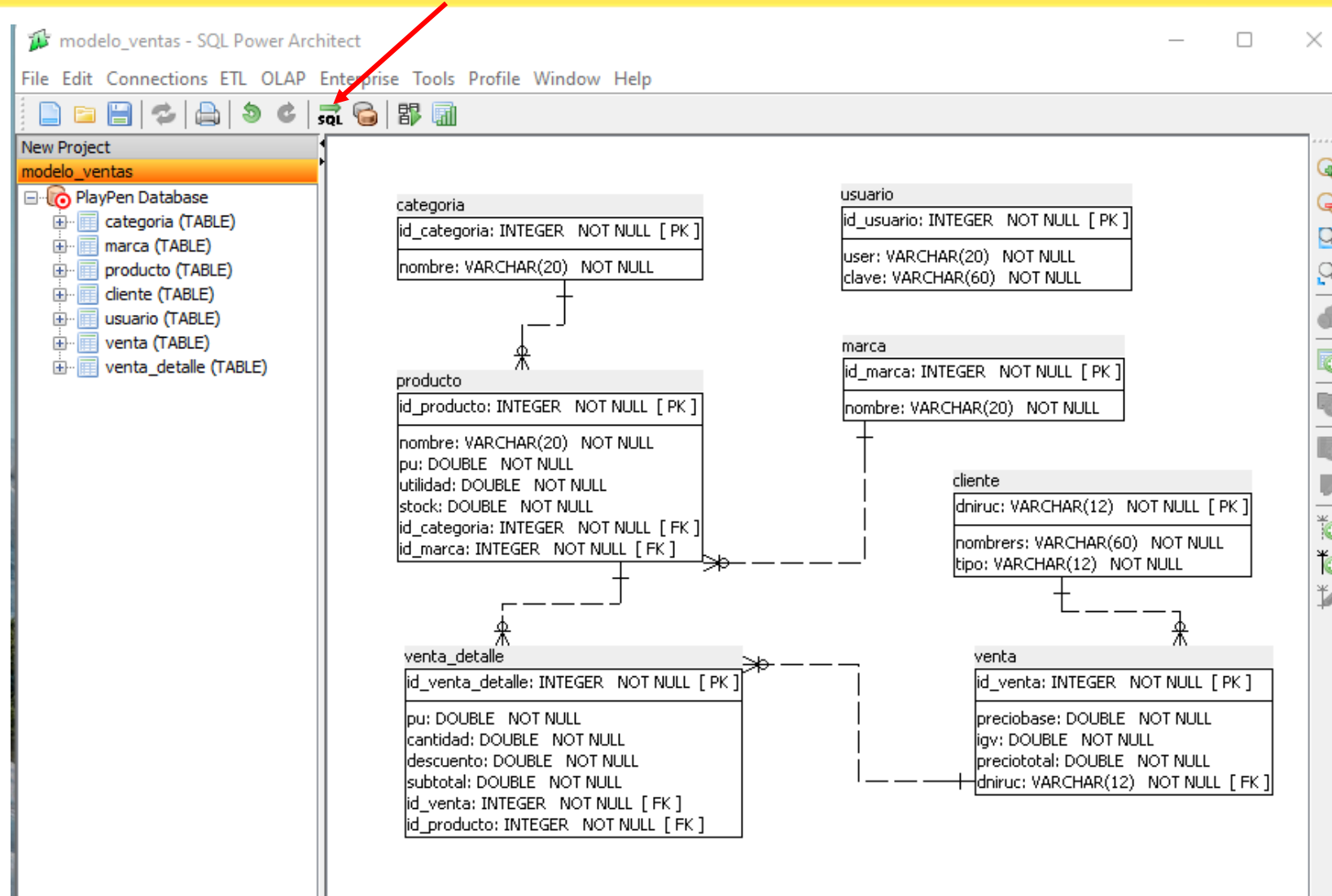
dniruc	Cliente	
43631971	nombreRS	Tipo
1.044E+10	Raul Porras	General
2.01E+10	Juan Bautista	Natural
	Universidad Peruana Unio	Juridica

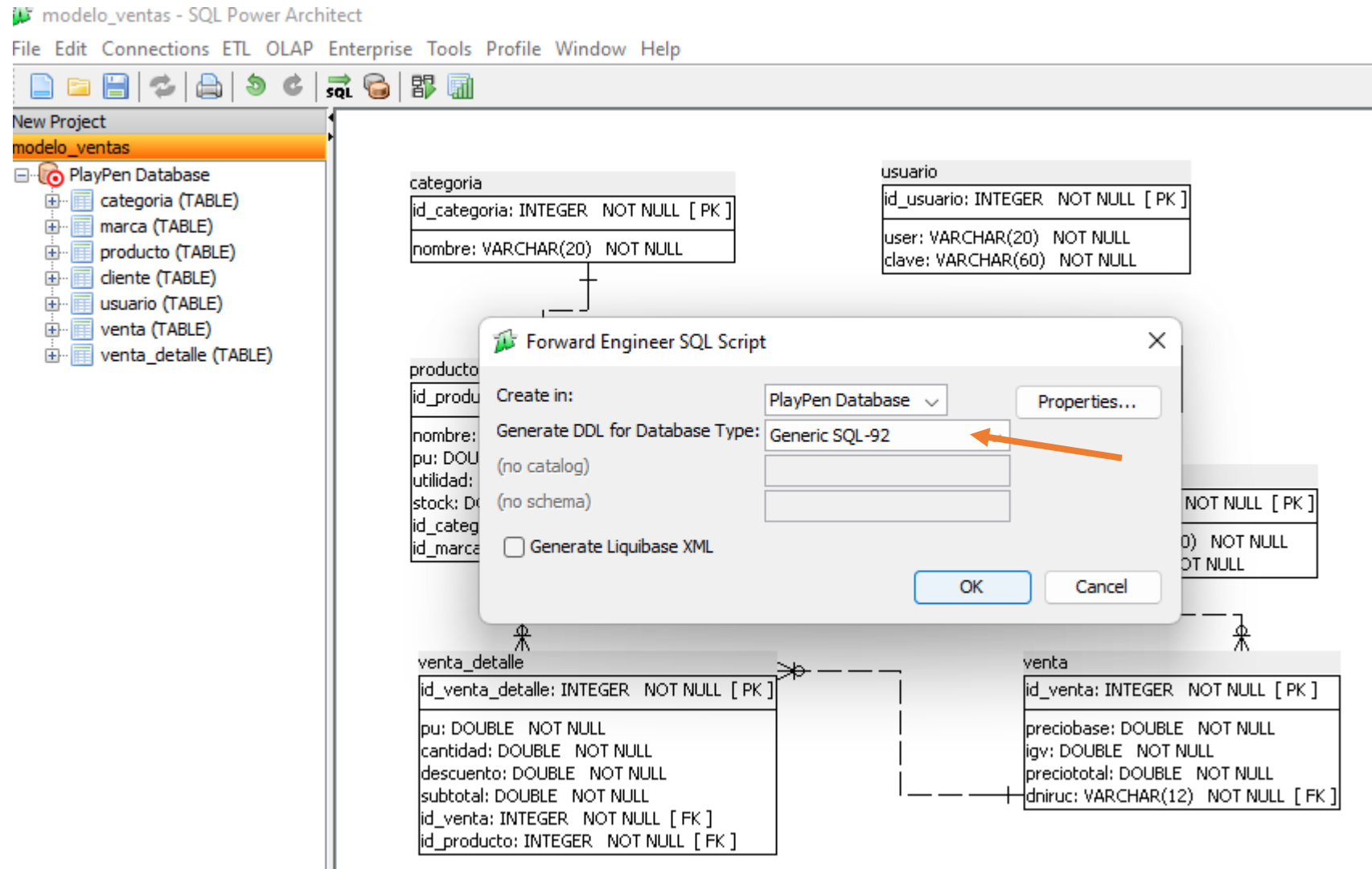
Producto						
IdProducto	Nombre	pu	utilidad	IdCate	IdMarca	stock
P001	Laptop 55454	2000	200	C001	M004	50
P002	Tarjeta Video 5454	200	20	C003	M005	30

Usuario	
user	clave
josemm	123456
frank	123456

Opcion 1	Venta			
IdVenta	dniruc	preciobase	igv	precioTotal
V001	20102121218	63140	11365.2	74505.2

VentaDetalle						
IdVentDet	IdVenta	IdProducto	descuento	pu	cantidad	subtotal
VD0001	V001	P001	3300	2200	30	62700
VD0002	V001	P002	0	220	2	440
VD0003	V002	P001	0	2200	1	2200





The screenshot displays the SQL Power Architect interface for a project named 'modelo\_ventas'. The left sidebar shows a 'New Project' list with 'modelo\_ventas' selected, and a 'PlayPen Database' containing tables: categoria (TABLE), marca (TABLE), producto (TABLE), cliente (TABLE), usuario (TABLE), venta (TABLE), and venta\_detalle (TABLE). The main workspace shows a partial database model with tables like 'categoria', 'producto', 'venta', and 'marca'. Overlaid on this is a 'Preview SQL Script' dialog box. An orange arrow points from the 'Preview SQL Script' dialog to the 'Forward Engineer SQL Script' dialog. The 'Preview SQL Script' dialog contains the following SQL code:

```
CREATE TABLE usuario (  
  id_usuario INTEGER NOT NULL,  
  user VARCHAR(20) NOT NULL,  
  clave VARCHAR(60) NOT NULL,  
  CONSTRAINT usuario_pk PRIMARY KEY  
  (id_usuario)  
);  
  
CREATE TABLE cliente (  
  dniruc VARCHAR(12) NOT NULL,  
  nombres VARCHAR(60) NOT NULL,  
  tipo VARCHAR(12) NOT NULL,  
  CONSTRAINT cliente_pk PRIMARY KEY (dniruc)  
);  
  
CREATE TABLE venta (  
  id_venta INTEGER NOT NULL,  
  preciobase DOUBLE NOT NULL,  
  igv DOUBLE NOT NULL,  
  preciototal DOUBLE NOT NULL,  
  dniruc VARCHAR(12) NOT NULL,  
  CONSTRAINT venta_pk PRIMARY KEY (id_venta)  
);  
  
CREATE TABLE marca (  
  id_marca INTEGER NOT NULL,  
  nombre VARCHAR(20) NOT NULL,  
  CONSTRAINT marca_pk PRIMARY KEY
```

The 'Forward Engineer SQL Script' dialog box is open, showing options to 'Create in: PlayPen Database' and 'Generate DDL for Database Type: Generic SQL-92'. It also includes checkboxes for '(no catalog)', '(no schema)', and 'Generate Liquibase XML'. The 'OK' button is highlighted.



```
CREATE TABLE usuario (  
    id_usuario INTEGER NOT NULL,  
    user VARCHAR(20) NOT NULL,  
    clave VARCHAR(60) NOT NULL,  
    CONSTRAINT usuario_pk PRIMARY KEY (id_usuario AUTOINCREMENT)  
);  
  
CREATE TABLE cliente (  
    dniruc VARCHAR(12) NOT NULL,  
    nombrers VARCHAR(60) NOT NULL,  
    tipo VARCHAR(12) NOT NULL,  
    CONSTRAINT cliente_pk PRIMARY KEY (dniruc)  
);  
  
CREATE TABLE venta (  
    id_venta INTEGER NOT NULL,  
    preciobase DOUBLE NOT NULL,  
    igv DOUBLE NOT NULL,  
    preciototal DOUBLE NOT NULL,  
    dniruc VARCHAR(12) NOT NULL,  
    CONSTRAINT venta_pk PRIMARY KEY (id_venta AUTOINCREMENT),  
                                     FOREIGN KEY("dniruc")  
REFERENCES "cliente"("dniruc") ON UPDATE RESTRICT ON DELETE RESTRICT  
);
```

```
CREATE TABLE marca (  
    id_marca INTEGER NOT NULL,  
    nombre VARCHAR(20) NOT NULL,  
    CONSTRAINT marca_pk PRIMARY KEY (id_marca  
AUTOINCREMENT)  
);  
  
CREATE TABLE categoria (  
    id_categoria INTEGER NOT NULL,  
    nombre VARCHAR(20) NOT NULL,  
    CONSTRAINT categoria_pk PRIMARY KEY (id_categoria  
AUTOINCREMENT)  
);
```

```
CREATE TABLE producto (  
    id_producto INTEGER NOT NULL,  
    nombre VARCHAR(20) NOT NULL,  
    pu DOUBLE NOT NULL,  
    utilidad DOUBLE NOT NULL,  
    stock DOUBLE NOT NULL,  
    id_categoria INTEGER NOT NULL,  
    id_marca INTEGER NOT NULL,  
    CONSTRAINT producto_pk PRIMARY KEY (id_producto  
AUTOINCREMENT),  
                                FOREIGN KEY  
(id_marca) REFERENCES marca (id_marca) ON DELETE NO ACTION  
ON UPDATE NO ACTION NOT DEFERRABLE,  
                                FOREIGN KEY  
(id_categoria) REFERENCES categoria (id_categoria) ON DELETE NO  
ACTION ON UPDATE NO ACTION NOT DEFERRABLE  
);
```

```
CREATE TABLE venta_detalle (  
    id_venta_detalle INTEGER NOT NULL,  
    pu DOUBLE NOT NULL,  
    cantidad DOUBLE NOT NULL,  
    descuento DOUBLE NOT NULL,  
    subtotal DOUBLE NOT NULL,  
    id_venta INTEGER NOT NULL,  
    id_producto INTEGER NOT NULL,  
    CONSTRAINT venta_detalle_pk PRIMARY KEY (id_venta_detalle  
AUTOINCREMENT),  
                                FOREIGN KEY (id_venta)  
REFERENCES venta (id_venta) ON DELETE NO ACTION ON UPDATE NO ACTION  
NOT DEFERRABLE,  
                                FOREIGN KEY (id_producto)  
REFERENCES producto (id_producto) ON DELETE NO ACTION ON UPDATE NO  
ACTION NOT DEFERRABLE  
);
```

```
BufferedImage image;
UtilsX obj=new UtilsX();
public Login() {
    initComponents();
    this.setTitle("Login SysCenterLife");
    try {
        image=ImageIO.read(obj.getFile("secrecy-icon.png"));
    } catch (Exception ex) {
        System.out.println(ex.getMessage());
    }
    this.setIconImage(image);
    this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    this.setResizable(false);
    Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
    this.setSize(new Dimension(screenSize.width/2, (screenSize.height-36)/2));
    this.setLocationRelativeTo(null);
    this.setVisible(true);
}
```

En acción perfomet

```
System.out.println("Entro Aqui!!");
GUIMain guiMain = new GUIMain();
guiMain.setVisible(true);
this.dispose();
```

```
public class GUIMain extends JFrame {  
  
    JMenuBar menuBar;  
    JMenu menu1;  
    JMenuItem jmI1;  
    JTabbedPane jtpane;  
    JPanel jp;  
    JScrollPane scrollPane;  
  
}
```

```
public GUIMain() {  
    this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();  
    this.setSize(new Dimension(screenSize.width, (screenSize.height) - 36));  
    menuBar = new JMenuBar();  
    menu1 = new JMenu("Archivo");  
    jmI1 = new JMenuItem("Abrir");  
    menuBar.add(menu1);  
    menu1.add(jmI1);  
    menu1 = new JMenu("Ver");  
    menuBar.add(menu1);  
    this.add(menuBar);  
  
    MenuItemListener menuItemListener = new MenuItemListener();  
    jmI1.addActionListener(menuItemListener);  
  
    this.getContentPane().add(BorderLayout.NORTH, menuBar);  
    //this.getContentPane().add(BorderLayout.CENTER, jtpane);  
    this.setVisible(true);  
}
```

```
class MenuItemListener implements ActionListener {
    @Override
    public void actionPerformed(ActionEvent e) {
        System.out.println("pasa por aqui");
        Container contai = GUIMain.this.getContentPane();
        if (e.getSource() == jmI1) {

            jtpane = new JTabbedPane();
            jp = new JPanel();
            jtpane.add("Prueba", jp);
            jp = new JPanel();
            jtpane.add("Prueba 2", jp);

            JPanel pp = new JPanel();
            pp.setPreferredSize(new Dimension(2000, 1000));
            JScrollPane = new JScrollPane(pp);
            JScrollPane.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);
            JScrollPane.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_ALWAYS);

            JPanel pp1 = new JPanel();
            pp1.setPreferredSize(new Dimension(2000, 1000));

            JPanel pp2 = new JPanel();
            pp2.setPreferredSize(new Dimension(2000, 1000));

            jtpane.setBounds(10, 20, 400, 200);
            jtpane.add("main", pp);
            jtpane.add("visit", pp1);
            jtpane.add("ver", pp2);

            jtpane.setTabLayoutPolicy(JTabbedPane.SCROLL_TAB_LAYOUT);
            contai.add(BorderLayout.CENTER, jtpane);
            contai.invalidate();
            contai.validate();
            contai.repaint();
        }
    }
}
```

## Dependencia

```
<!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
<dependency>
  <groupId>org.projectlombok</groupId>
  <artifactId>lombok</artifactId>
  <version>1.18.24</version>
  <scope>provided</scope>
</dependency>
```

```
import lombok.Data;

@Data
public class ClienteTO {
    public String dniruc, nombresrs, tipo;
}
```

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import javax.swing.JOptionPane;

public class Conn {
    public static final String DEFAULT_DATE_STRING_FORMAT_PE = "dd/MM/yyyy";
    public static final String DEFAULT_DATE_STRING_FORMAT = "yyyy-MM-dd HH:mm:ss";
    static Connection conn = null;

    public static Connection connectSQLite() {
        try {
            Class.forName("org.sqlite.JDBC");
            String dbURL = "jdbc:sqlite:data/ventasdb.db?foreign_keys=on;";
            if (conn == null)
                conn = DriverManager.getConnection(dbURL);
            System.out.println("Conexion Exitosa");
        } catch (ClassNotFoundException | SQLException e) {
            JOptionPane.showMessageDialog(null, "Error en la conexión" + e);
        }
        return conn;
    }

    public static void closeSQLite(Connection conn) {
        try {
            if (conn != null) { conn.close(); }
        } catch (SQLException ex) {
            ex.printStackTrace();
        }
    }
}
```

```
public interface ClienteDaoI {  
    public int create(ClienteTO d);  
    public int update(ClienteTO d);  
    public int delete(String id) throws Exception;  
    public List<ClienteTO> listCmb(String filter);  
    public List listarClientes();  
    public ClienteTO buscarClientes(String dni);  
    public void reportarCliente();  
}
```

```
public class ClienteDAO implements ClienteDaoI{  
  
    Statement stmt = null;  
    Vector columnNames;  
    Vector visitdata ;  
    Connection connection = Conn.connectSQLite();  
    static PreparedStatement ps;  
    static ErrorLogger log = new ErrorLogger(ClienteDAO.class.getName());  
    ResultSet rs = null;  
  
    public ClienteDAO() {  
        columnNames = new Vector();  
        visitdata = new Vector();  
    }  
}
```



```
public int create(ClienteTO d) {
    int rsId = 0;
    String[] returns = {"dniruc"};
    String sql = "INSERT INTO cliente(dniruc, nombres, tipo) "
        + "VALUES(?,?,?)";
    int i = 0;
    try {
        ps = connection.prepareStatement(sql, returns);
        ps.setString(++i, d.getDniruc());
        ps.setString(++i, d.getNombresrs());
        ps.setString(++i, d.getTipo());
        rsId = ps.executeUpdate();// 0 no o 1 si commit
        try (ResultSet rs = ps.getGeneratedKeys()) {
            if (rs.next()) {
                rsId = rs.getInt(1);
            }
            rs.close();
        }
    } catch (SQLException ex) {
        //System.err.println("create:" + ex.toString());
        log.log(Level.SEVERE, "create", ex);
    }
    return rsId;
}
```

```
public int update(ClienteTO d) {
    System.out.println("actualizar d.getDniruc: " + d.getDniruc());
    int comit = 0;
    String sql = "UPDATE cliente SET "
        + "nombres=?, "
        + "tipo=? "
        + "WHERE dniruc=?";
    int i = 0;
    try {
        ps = connection.prepareStatement(sql);
        ps.setString(++i, d.getNombresrs());
        ps.setString(++i, d.getTipo());
        ps.setString(++i, d.getDniruc());
        comit = ps.executeUpdate();

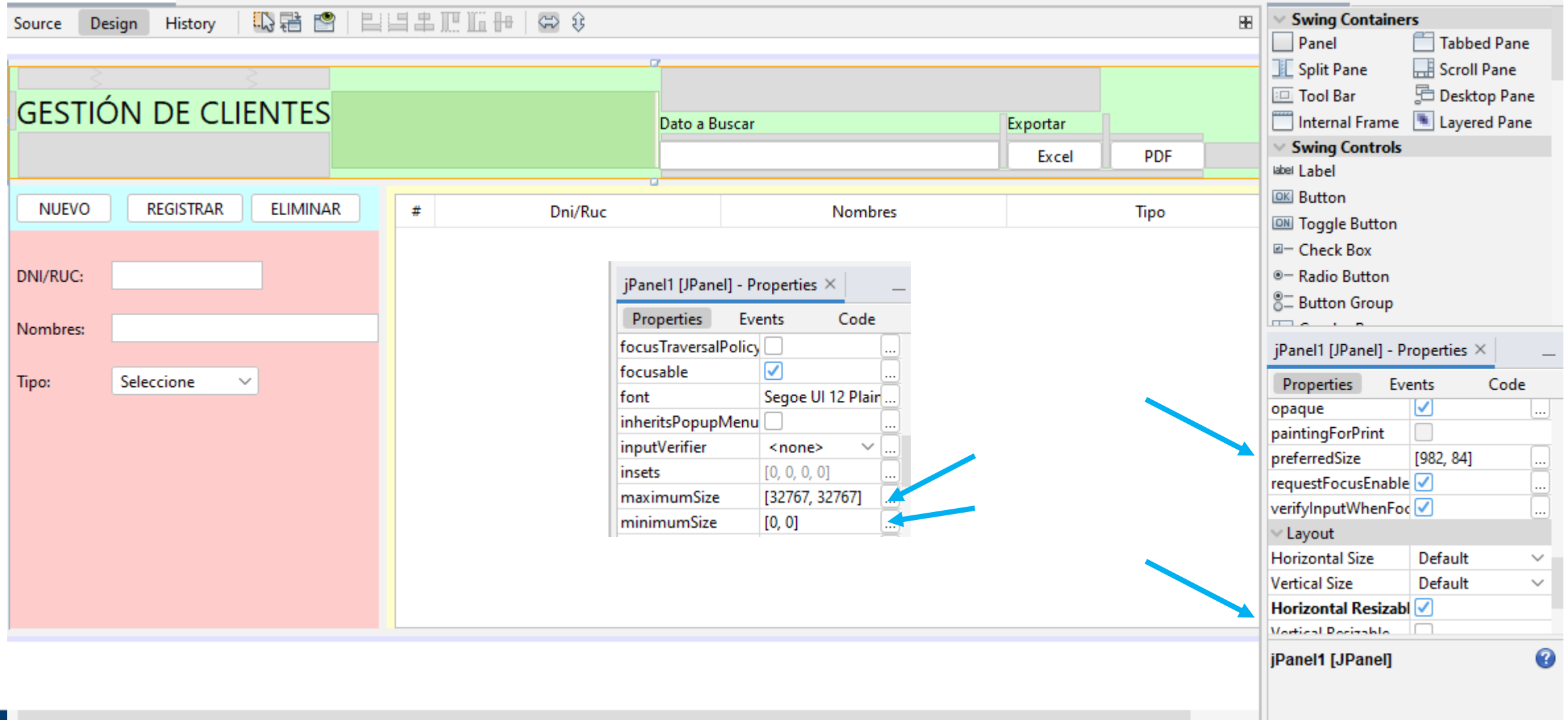
    } catch (SQLException ex) {
        log.log(Level.SEVERE, "update", ex);
    }
    return comit;
}
```

```
public int delete(String id) throws Exception {
    int comit = 0;
    String sql = "DELETE FROM cliente WHERE dniruc = ?";
    try {
        ps = connection.prepareStatement(sql);
        ps.setString(1, id);
        comit = ps.executeUpdate();
    } catch (SQLException ex) {
        log.log(Level.SEVERE, "delete", ex);
        // System.err.println("NO del " + ex.toString());
        throw new Exception("Detalle:" + ex.getMessage());
    }
    return comit;
}
```

```
public List listarClientes(){
    List<ClienteTO> listarclientes = new ArrayList();
    String sql = "SELECT * FROM cliente";
    try {
        connection = new Conn().connectSQLite();
        ps = connection.prepareStatement(sql);
        rs = ps.executeQuery();
        while (rs.next()) {
            ClienteTO cli = new ClienteTO();
            cli.setDniruc(rs.getString("dniruc"));
            cli.setNombresrs(rs.getString("nombresrs"));
            cli.setTipo(rs.getString("tipo"));
            listarclientes.add(cli);
        }
    } catch (SQLException e) {
        System.out.println(e.toString());
    }
    return listarclientes;
}
```

```
public List<ClienteTO> listCmb(String filter) {  
    List<ClienteTO> ls = new ArrayList();  
    ls.add(new ClienteTO());  
    ls.addAll(listarClientes());  
    return ls;  
}
```

```
public ClienteTO buscarClientes(String dni){  
    ClienteTO cliente = new ClienteTO();  
    String sql = "SELECT * FROM cliente WHERE dniruc = ?";  
    try {  
        connection = new Conn().connectSQLite();  
        ps = connection.prepareStatement(sql);  
        ps.setString(1, dni);  
        rs = ps.executeQuery();  
        if (rs.next()) {  
            cliente.setDniruc(rs.getString("dniruc"));  
            cliente.setNombres(rs.getString("nombres"));  
            cliente.setTipo(rs.getString("tipo"));  
        }  
    } catch (SQLException e) {  
        System.out.println(e.toString());  
    }  
    return cliente;  
}
```



Source Design History

GESTIÓN DE CLIENTES

Dato a Buscar Exportar Excel PDF

NUEVO REGISTRAR ELIMINAR

DNI/RUC:

Nombres:

Tipo: Seleccione ▼

#	Dni/Ruc	Nombres	Tipo

jPanel1 [JPanel] - Properties

Properties	Events	Code
focusTraversalPolicy	<input type="checkbox"/>	...
focusable	<input checked="" type="checkbox"/>	...
font	Segoe UI 12 Plair	...
inheritsPopupMenu	<input type="checkbox"/>	...
inputVerifier	<none>	...
insets	[0, 0, 0, 0]	...
maximumSize	[32767, 32767]	...
minimumSize	[0, 0]	...

jPanel1 [JPanel] - Properties

Properties	Events	Code
opaque	<input checked="" type="checkbox"/>	...
paintingForPrint	<input type="checkbox"/>	...
preferredSize	[982, 84]	...
requestFocusEnable	<input checked="" type="checkbox"/>	...
verifyInputWhenFoc	<input checked="" type="checkbox"/>	...
Layout		
Horizontal Size	Default	▼
Vertical Size	Default	▼
Horizontal Resizable	<input checked="" type="checkbox"/>	...
Vertical Resizable	<input type="checkbox"/>	...

jPanel1 [JPanel]

The screenshot displays a GUI design tool interface for a 'GESTIÓN DE CLIENTES' (Customer Management) application. The main window features a header with the title and search/export options, a table with columns for ID, DNI/RUC, Names, and Type, and a form on the left for adding new clients. Annotations highlight the 'NUEVO' button and its associated properties and code generation settings.

**Annotations:**

- A blue arrow points from the 'NUEVO' button to the 'Events' tab of the 'jButton1 [JButton] - Properties' window.
- A green arrow points from the 'NUEVO' button to the 'Code Generation' tab of the 'jButton1 [JButton] - Properties' window.
- A green arrow points from the 'NUEVO' button to the 'Code Generation' tab of the 'jButton1 [JButton] - Properties' window.

**GUI Elements:**

- Header:** GESTIÓN DE CLIENTES, Dato a Buscar, Exportar (Excel, PDF).
- Buttons:** NUEVO, REGISTRAR, ELIMINAR.
- Form:** DNI/RUC, Nombres, Tipo (Seleccione).
- Table:** #, Dni/Ruc, Nombres, Tipo.

**Properties Windows:**

- jButton1 [JButton] - Properties:**
  - Events:** actionPerformed (jButton1..., java.awt.event.ActionListener), ancestorAdded (<none>), ancestorMoved (<none>), ancestorRemoved (<none>), ancestorResized (<none>), caretPositionChange (<none>), componentAdded (<none>).
  - Code Generation:** Bean Class (class javax.swing...), Variable Name (jButton1), Variable Modifiers (private), Type Parameters, Use Local Variable (private), Generate Mnemoni, Custom Creation C, Pre-Creation Code.

## GESTIÓN DE CLIENTES

Dato a Buscar
Exportar
Excel
PDF

NUEVO
REGISTRAR
ELIMINAR

DNI/RUC:
Nombres:
Tipo: Seleccione

#	Dni/Ruc	Nombres	Tipo

jTable1 [JTable] - model

Set jTable1's model property using: Table model customizer

Table Model

Table Settings Default Values

Specify Title and Column Types Here:

Column	Title	Type	Editable
1	#	Object	<input checked="" type="checkbox"/>
2	Dni/Ruc	Object	<input checked="" type="checkbox"/>
3	Nombres	Object	<input checked="" type="checkbox"/>
4	Tipo	Object	<input checked="" type="checkbox"/>

Insert  
Delete  
Move Up  
Move Down

Tool Bar Desktop Pane  
Internal Frame Layered Pane

Swing Controls

- Label
- Button
- Toggle Button
- Check Box
- Radio Button
- Button Group

jTable1 [JTable] - Properties

Properties	Events	Code
autoCreateColumn	<input checked="" type="checkbox"/>	...
autoCreateRowSort	<input type="checkbox"/>	...
background	<input type="checkbox"/> [255,255,255]	...
border	(No Border)	...
font	Segoe UI 12 Plain	...
foreground	<input checked="" type="checkbox"/> [0,0,0]	...
model	[TableModel]	...
toolTipText		...

```
enum TIPOCLXIENTE {  
    Natural, General, Juridico  
};
```

```
public void ListarClientes() {  
    cDao = new ClienteDAO();  
    List<ClienteTO> listarCleintes = cDao.listarClientes();  
    jTable1.setAutoCreateRowSorter(true);  
    modelo = (DefaultTableModel) jTable1.getModel();  
    Object[] ob = new Object[4];  
    for (int i = 0; i < listarCleintes.size(); i++) {  
        ob[0] = i + 1;  
        ob[1] = listarCleintes.get(i).getDniruc();  
        ob[2] = listarCleintes.get(i).getNombresrs();  
        ob[3] = listarCleintes.get(i).getTipo();  
        modelo.addRow(ob);  
    }  
    jTable1.setModel(modelo);  
}
```

```
ClienteDaoI cDao;  
DefaultTableModel modelo;  
  
public ClienteMain() {  
    initComponents();  
    ListarClientes();  
    for (TIPOXCLIENTE myVar : TIPOXCLIENTE.values()) {  
        cbxTipo.addItem(myVar.toString());  
    }  
}
```

```
private void paintForm() {  
    if (jTable1.getSelectedRow() != -1) {  
        modelo = (DefaultTableModel) jTable1.getModel();  
        int rowx = jTable1.getSelectedRow();  
        Object valor = jTable1.getValueAt(rowx, 1);  
        //ClienteTO filax = (ClienteTO)  
        modelo.getRow(jTable1.getSelectedRow());  
        cDao = new ClienteDAO();  
        ClienteTO d =  
        cDao.buscarClientes(valor.toString());  
        txtDNI.setText(d.getDniruc());  
        txtNonbre.setText(d.getNombresrs());  
        cbxTipo.setSelectedItem(d.getTipo());  
        txtDNI.setEditable(false);  
        jButton2.setText("MODIFICAR");  
        //guardarButton.setToolTipText("MODIFICAR");  
    } else {  
        txtDNI.setEditable(true);  
    }  
}
```

```
public void resetForm() {  
    txtDNI.setText("");  
    txtNonbre.setText("");  
    cbxTipo.setSelectedIndex(0);  
    txtDNI.requestFocus();  
}
```



```
cDao = new ClienteDAO();
ClienteTO to = new ClienteTO();
to.setDniruc(txtDNI.getText());
to.setNombresrs(txtNonbre.getText());
to.setTipo(cbxTipo.getSelectedItem().toString());
int fila = jTable1.getSelectedRow();
if (fila != -1) {
    try {
        int resultado = cDao.update(to);
        if (resultado != 0) {
            modelo = (DefaultTableModel) jTable1.getModel();
            Object nuevo[] = {fila + 1, to.getDniruc(), to.getNombresrs(), to.getTipo()};
            modelo.removeRow(fila);
            modelo.insertRow(fila, nuevo);
            resetForm();
            JOptionPane.showMessageDialog(this, "Re registro");
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
    }
} else {
    try {
        if (cDao.create(to) != 0) {
            modelo = (DefaultTableModel) jTable1.getModel();
            Object nuevo[] = {modelo.getRowCount() + 1, to.getDniruc(), to.getNombresrs(), to.getTipo()};
            modelo.addRow(nuevo);
            resetForm();
            JOptionPane.showMessageDialog(this, "Re registro");
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
    }
}
```

```
cDao = new ClienteDAO();  
if (jTable1.getSelectedRowCount() > 0) {  
    try {  
        modelo = (DefaultTableModel) jTable1.getModel();  
        int rowx = jTable1.getSelectedRow();  
        Object valor = jTable1.getValueAt(rowx, 1);  
        JOptionPane.showMessageDialog(this, valor);  
        modelo.removeRow(rowx);  
        cDao.delete(valor.toString());  
        resetForm();  
    } catch (Exception e) {  
        JOptionPane.showMessageDialog(this, e.getMessage());  
    }  
}  
  
} else {  
    JOptionPane.showMessageDialog(this, "Seleccione un item");  
}
```

```
paintForm();
```

Botón Nuevo

```
resetForm();  
jButton2.setText("REGISTRAR");  
txtDNI.setEditable(true);  
jTable1.getSelectionModel().clearSelection();
```



# MUCHAS GRACIAS

*Sé Íntegro, Sé Misionero, Sé Innovador*

---

[www.upeu.edu.pe](http://www.upeu.edu.pe)