

# TALLER DE DESARROLLO DE APLICACIONES 1

**SEMANA: 4**

**ENTREGADO POR:** Mescua Segovia Marino

**CÓDIGO:** N04058C

**CATEDRÁTICO:** Fernández Bejarano Raúl Enrique

## **ACTIVIDAD 1:**

### **DIAGRAMA DE CLASES:**



### **DISEÑO DE LA APLICACIÓN:**

### LOGIN

USUARIO:

CONTRASEÑA:

.....

ENTRAR

REGISTRAR

SALIR

CALCULAR   SALIR

INGRESAR LADO:

AREA:

CALCULAR

NUEVO

CALCULARSALIR

INGRESE LADO:

INGRESE BASE:

AREA:

CALCULAR

NUEVO

REGISTRAR

DNI:

NOMBRE:

APELLIDO:

CONTRASEÑA:

REPETIR CONTRASEÑA:

CREAR

SALIR

**CÓDIGO DE LA APLICACIÓN:**

```

5 public class LoginControlador {
6
7     LoginVista vista;
8
9     public LoginControlador(LoginVista vista) {
10         this.vista = vista;
11
12         vista.btnEntrar.addActionListener(e -> ingresar());
13         vista.btnSalir.addActionListener(e -> salir());
14         vista.btnRegistrar.addActionListener(e -> abrirRegistro());
15     }
16
17     public void ingresar() {
18         String usuarioIngresado = vista.txt_usuario.getText();
19         String contrasenaIngresada = new String(vista.txtContrasena.getPassword());
20
21         if (usuarioIngresado.equals("admin") && contrasenaIngresada.equals("admin")) {
22             JOptionPane.showMessageDialog(null, "Bienvenido Admin");
23             // Abrir MenuAdmin aquí
24         } else {
25             boolean encontrado = false;
26
27             for (ModeloUsuario u : ModeloUsuario.listaUsuarios) {
28                 if (u.dni.equals(usuarioIngresado) && u.contrasena.equals(contrasenaIngresada)) {
29                     JOptionPane.showMessageDialog(null, "Bienvenido " + u.nombre);
30                     // Abrir Menu Usuario aquí
31                     encontrado = true;
32                     break;
33                 }
34             }
35
36             if (!encontrado) {
37                 JOptionPane.showMessageDialog(null, "Usuario o contraseña incorrectos");
38             }
39         }
40     }
41
42     public void salir() {
43         vista.dispose();
44     }
45
46     public void abrirRegistro() {
47         JOptionPane.showMessageDialog(null, "Aquí abrirías el formulario de registro");
48         // Puedes hacer new RegistroVista() y ControladorRegistro
49     }
50 }

```

```

12 public class MenuAdminControlador {
13
14     MenuAdmin vista;
15     String figuraSeleccionada = "";
16
17     public MenuAdminControlador(MenuAdmin vista) {
18         this.vista = vista;
19
20         // Menú
21         vista.itemCuadrado.addActionListener(e -> seleccionarFigura("cuadrado"));
22         vista.itemCirculo.addActionListener(e -> seleccionarFigura("circulo"));
23         vista.itemRectangulo.addActionListener(e -> seleccionarFigura("rectangulo"));
24         vista.itemTriangulo.addActionListener(e -> seleccionarFigura("triangulo"));
25
26         // Botones
27         vista.btn_calcular.addActionListener(e -> calcular());
28         vista.btn_nuevo.addActionListener(e -> nuevo());
29
30         vista.btn_salir.addActionListener(e -> vista.dispose());
31
32         // Escritura en campos
33         vista.txt_lado.addKeyListener(new KeyAdapter() {
34             public void keyReleased(KeyEvent e) {
35                 validarInputs();
36             }
37         });
38
39         vista.txt_base.addKeyListener(new KeyAdapter() {
40             public void keyReleased(KeyEvent e) {
41                 validarInputs();
42             }
43         });
44     }
45
46     public void seleccionarFigura(String figura) {
47         figuraSeleccionada = figura;
48         vista.txt_lado.setText("");
49         vista.txt_base.setText("");
50         vista.txt_area.setText("");
51
52         vista.txt_lado.setEnabled(false);
53         vista.txt_base.setEnabled(false);
54         vista.btn_calcular.setEnabled(false);
55         vista.btn_nuevo.setEnabled(false);
56
57         // Habilitar solo lo necesario
58         if (figura.equals("cuadrado") || figura.equals("circulo")) {

```

```

58     if (figura.equals("cuadrado") || figura.equals("circulo")) {
59         vista.txt_lado.setEnabled(true);
60     } else {
61         vista.txt_lado.setEnabled(true);
62         vista.txt_base.setEnabled(true);
63     }
64 }
65
66 public void validarInputs() {
67     boolean campoLleno = !vista.txt_lado.getText().isEmpty();
68     boolean campoBase = !vista.txt_base.getText().isEmpty();
69
70     if (figuraSeleccionada.equals("cuadrado") || figuraSeleccionada.equals("circulo")) {
71         vista.btn_calcular.setEnabled(campoLleno);
72     } else {
73         vista.btn_calcular.setEnabled(campoLleno && campoBase);
74     }
75 }
76
77 public void calcular() {
78     try {
79         double lado = Double.parseDouble(vista.txt_lado.getText());
80         double base = vista.txt_base.isEnabled() ? Double.parseDouble(vista.txt_base.getText()) : 0;
81         double area = 0;
82
83         switch (figuraSeleccionada) {
84             case "cuadrado":
85                 area = lado * lado;
86                 break;
87             case "circulo":
88                 area = Math.PI * lado * lado;
89                 break;
90             case "rectangulo":
91                 area = lado * base;
92                 break;
93             case "triangulo":
94                 area = (lado * base) / 2;
95                 break;
96         }
97
98         vista.txt_area.setText(String.valueOf(area));
99         vista.txt_lado.setEnabled(false);
100        vista.txt_base.setEnabled(false);
101        vista.btn_calcular.setEnabled(false);
102        vista.btn_nuevo.setEnabled(true);
103    } catch (NumberFormatException e) {
104        JOptionPane.showMessageDialog(null, "Ingrese números válidos");
105    }
106 }
107
108
109 public void nuevo() {
110     vista.txt_lado.setText("");
111     vista.txt_base.setText("");
112     vista.txt_area.setText("");
113     vista.btn_calcular.setEnabled(false);
114     vista.btn_nuevo.setEnabled(false);
115
116     if (figuraSeleccionada.equals("cuadrado") || figuraSeleccionada.equals("circulo")) {
117         vista.txt_lado.setEnabled(true);
118     } else {
119         vista.txt_lado.setEnabled(true);
120         vista.txt_base.setEnabled(true);
121     }
122 }
123
124

```

```

12 public class MenuControlador {
13
14     Menu vista;
15     String figuraSeleccionada = "";
16
17     public MenuControlador(Menu vista) {
18         this.vista = vista;
19
20         // Asociar eventos
21         vista.itemCuadrado.addActionListener(e -> seleccionarFigura("cuadrado"));
22         vista.itemCirculo.addActionListener(e -> seleccionarFigura("circulo"));
23
24         vista.btnCalcular.addActionListener(e -> calcular());
25         vista.btnNuevo.addActionListener(e -> nuevo());
26
27         vista.txt_lado.addKeyListener(new KeyAdapter() {
28             public void keyReleased(KeyEvent e) {
29                 if (!vista.txt_lado.getText().isEmpty()) {
30                     vista.btnCalcular.setEnabled(true);
31                 } else {
32                     vista.btnCalcular.setEnabled(false);
33                 }
34             }
35         });
36     }
37
38     public void seleccionarFigura(String figura) {
39         figuraSeleccionada = figura;
40         vista.txt_lado.setText("");
41         vista.txt_area.setText("");
42         vista.txt_lado.setEnabled(true);
43         vista.btnCalcular.setEnabled(false);
44         vista.btnNuevo.setEnabled(false);
45     }
46
47     public void calcular() {
48         try {
49             double valor = Double.parseDouble(vista.txt_lado.getText());
50             double area = 0;
51
52             if (figuraSeleccionada.equals("cuadrado")) {
53                 area = valor * valor;
54             } else if (figuraSeleccionada.equals("circulo")) {
55                 area = Math.PI * valor * valor;
56             }
57
58             vista.txt_area.setText(String.valueOf(area));
59             vista.txt_lado.setEnabled(false);
60             vista.btnCalcular.setEnabled(false);
61             vista.btnNuevo.setEnabled(true);
62         } catch (NumberFormatException e) {
63             JOptionPane.showMessageDialog(null, "Ingrese un valor válido");
64         }
65     }
66

```

```

66
67 public void nuevo() {
68     vista.txt_lado.setText("");
69     vista.txt_area.setText("");
70     vista.txt_lado.setEnabled(true);
71     vista.btnCalcular.setEnabled(false);
72     vista.btnNuevo.setEnabled(false);
73 }
74 }
75

```

```

15 public class RegistroControlador {
16
17     RegistroVista vista;
18
19     public RegistroControlador(RegistroVista vista) {
20         this.vista = vista;
21
22         vista.btnCrear.addActionListener(e -> registrarUsuario());
23     }
24
25     public void registrarUsuario() {
26         String dni = vista.txtDni.getText();
27         String nombre = vista.txtNombre.getText();
28         String apellido = vista.txtApellido.getText();
29         String contrasena = new String(vista.txtContrasena.getPassword());
30         String repetir = new String(vista.txtRepetirContrasena.getPassword());
31
32         // Validaciones
33         if (dni.isEmpty() || nombre.isEmpty() || apellido.isEmpty() || contrasena.isEmpty() || repetir.isEmpty()) {
34             JOptionPane.showMessageDialog(null, "Complete todos los campos");
35             return;
36         }
37
38         if (!contrasena.equals(repetir)) {
39             JOptionPane.showMessageDialog(null, "Las contraseñas no coinciden");
40             return;
41         }
42
43         for (ModeloUsuario u : ModeloUsuario.listaUsuarios) {
44             if (u.dni.equals(dni)) {
45                 JOptionPane.showMessageDialog(null, "DNI ya registrado");
46                 return;
47             }
48         }
49
50         ModeloUsuario nuevo = new ModeloUsuario(dni, nombre, apellido, contrasena);
51         ModeloUsuario.listaUsuarios.add(nuevo);
52         JOptionPane.showMessageDialog(null, "Usuario registrado correctamente");
53     }
54 }
55

```

```

12
13 public class Main {
14     public static void main(String[] args) {
15         LoginVista vista = new LoginVista();
16
17         vista.setVisible(true);
18     }
19 }
20

```



```

public class ModeloUsuario {
    public String dni;
    public String nombre;
    public String apellido;
    public String contrasena;

    public ModeloUsuario(String dni, String nombre, String apellido, String contrasena) {
        this.dni = dni;
        this.nombre = nombre;
        this.apellido = apellido;
        this.contrasena = contrasena;
    }

    public static java.util.ArrayList<ModeloUsuario> listaUsuarios = new java.util.ArrayList<>();
}

```

```

13 public class LoginVista extends javax.swing.JFrame {
14
15     /**
16      * Creates new form Login
17      */
18     public LoginVista() {
19         initComponents();
20         new LoginControlador(this);
21     }
22
23     /**
24      * This method is called from within the constructor to initialize the form.
25      * WARNING: Do NOT modify this code. The content of this method is always
26      * regenerated by the Form Editor.
27      */
28     @SuppressWarnings("unchecked")
29     Generated Code
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116     /**
117      * @param args the command line arguments
118      */
119     public static void main(String args[]) {
120         /* Set the Nimbus look and feel */
121         Look and feel setting code (optional)
122         //</editor-fold>
123
124         /* Create and display the form */
125         java.awt.EventQueue.invokeLater(new Runnable() {
126             public void run() {
127                 new LoginVista().setVisible(true);
128             }
129         });
130     }

```

```

- /**
-  *
-  * @author marino
-  */
- public class Menu extends javax.swing.JFrame {
-
-     /**
-      * Creates new form FMenu
-      */
-     public Menu() {
-         initComponents();
-         new MenuControlador(this);
-     }
-
-     /**
-      * This method is called from within the constructor to initialize the form.
-      * WARNING: Do NOT modify this code. The content of this method is always
-      * regenerated by the Form Editor.
-      */
-     @SuppressWarnings("unchecked")
+     Generated Code
-
-     private void txt_areaActionPerformed(java.awt.event.ActionEvent evt) {
-         // TODO add your handling code here:
-     }
-
-     private void btnCalcularActionPerformed(java.awt.event.ActionEvent evt) {
-         // TODO add your handling code here:
-     }
-
-     private void btnNuevoActionPerformed(java.awt.event.ActionEvent evt) {
-         // TODO add your handling code here:
-     }
-
-     /**

```

```

public class MenuAdmin extends javax.swing.JFrame {

    /**
     * Creates new form MenuAdmin
     */
    public MenuAdmin() {
        initComponents();
        new MenuAdminController(this);
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    Generated Code

    private void txt_ladoActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void btn_calcularActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        Look and feel setting code (optional)

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new MenuAdmin().setVisible(true);
            }
        });
    }
}

```

```

13 public class RegistroVista extends javax.swing.JFrame {
14
15     /**
16      * Creates new form Registro
17      */
18     public RegistroVista() {
19         initComponents();
20         new RegistroControlador(this);
21     }
22
23     /**
24      * This method is called from within the constructor to initialize the form.
25      * WARNING: Do NOT modify this code. The content of this method is always
26      * regenerated by the Form Editor.
27      */
28     @SuppressWarnings("unchecked")
29     Generated Code
145
146     /**
147      * @param args the command line arguments
148      */
149     public static void main(String args[]) {
150         /* Set the Nimbus look and feel */
151         Look and feel setting code (optional)
152         //</editor-fold>
153
154         /* Create and display the form */
155         java.awt.EventQueue.invokeLater(new Runnable() {
156             public void run() {
157                 new RegistroVista().setVisible(true);
158             }
159         });
160     }

```