Tecnológico Nacional de México
Campus Orizaba

Tópicos Avanzados de Programación

Ingeniería en Sistemas Computacionales

Tema 1:
Interfaz gráfica de Usuario

Integrantes:

Muñoz Hernández Vania Lizeth – 21011009
Romero Ovando Karyme Michelle – 21011037
Maciel Villa Valeria – 21010985

Grupo:
4g2A

Fecha de entrega: 29 / Mayo / 2023

1. Introducción

La GUI es un componente esencial en muchas aplicaciones, ya que permite a los usuarios interactuar de manera intuitiva con el software, utilizando botones, cuadros de texto, menús desplegables y otros elementos visuales.

JFrame es una clase en Java que forma parte de la biblioteca Swing, la cual proporciona un conjunto de herramientas para construir interfaces gráficas. Con JFrame, se pueden crear ventanas en las que se mostrarán los componentes de la GUI. Estas ventanas pueden contener botones, etiquetas, campos de texto y otros elementos interactivos.

Los componentes de JFrame son elementos visuales que se pueden agregar a la ventana y se utilizan para capturar la entrada del usuario, mostrar información, realizar cálculos y muchas otras funciones.

En resumen, los tópicos avanzados de programación abren nuevas posibilidades para los desarrolladores al permitirles crear aplicaciones más interactivas y visualmente atractivas. El uso de JFrame y sus componentes en la construcción de interfaces gráficas es una de esas áreas avanzadas, brindando a los desarrolladores las herramientas necesarias para crear ventanas y agregar elementos interactivos que mejoren la experiencia del usuario.

2. Competencia específica

Desarrolla soluciones de software para resolver problemas en diversos contextos utilizando programación concurrente, acceso a datos, que soporten interfaz gráfica de usuario y consideren dispositivos móviles.

3. Material y Equipo

El material y equipo que se necesita para llevar a cabo la práctica son:

- ✓ Computadora
- ✓ Software y versión usados
- ✓ Materiales de apoyo para el desarrollo de la práctica

4. Desarrollo de la Práctica

Clase	Método	Descripción
ActionEvent		Representa una acción del usuario en la interfaz.
ChangeListener	ChangeListener	Recibe notificaciones cuando cambia el valor de una propiedad.
	stateChanged(ChangeEvent e)	Se invoca cuando el destino del listener ha cambiado de estado.

Clase	Método	Descripción
PlotOrientation		Se utiliza para indicar la orientación (horizontal o vertical) de un gráfico 2D.
ChartFactory		Una colección de métodos de utilidad para crear algunos gráficos estándar con JFreeChart.
DefaultCategoryDataset	DefaultCategory Dataset ()	Crea un nuevo conjunto de datos (vacío).
ChartPanel	ChartPanel()	El panel se registra en el gráfico para recibir notificaciones de cambios en cualquier componente del gráfico.
JFreeChart		Es una librería que ofrece la posibilidad de crear todo tipo de gráficas de manera sencilla y es de código abierto.
	getCategoryPlot()	Devuelve el reparto de la trama como un archivo CategoryPlot.
	getRenderer()	Devuelve el representador para el conjunto de datos principal.

	setSeriesPaint()	Establece la pintura utilizada para un relleno de serie y envía un mensaje RendererChangeEventa todos los oyentes registrados.
	getRangeAxis()	Devuelve el eje de rango del gráfico.
ChartFrame		Un marco para mostrar un gráfico.

Clase	Método	Descripción
JRadioButton	JRadioButton()	Constructor de la clase ButtonGroup.
		Permite seleccionar solo una opción de un conjunto de opciones.
	setBackground()	Establece el color de fondo de este componente.
	setText()	Establece el texto del botón.
	setBounds()	Mueve y cambia el tamaño de este componente.
	addChangeListener()	Agrega un ChangeListener al botón.
	add()	Agrega el menú emergente especificado al componente.

Clase	Método	Descripción
JCheckBox	JCheckBox()	Constructor de la clase JCheckBox.
	addItemListener()	Se conecta el componente con un objeto de la clase que maneja los sucesos originados en dicho componente.
	add()	Agrega el menú emergente especificado al componente.

	isSelected()	Devuelve true si el componente está seleccionado, en caso contrario devuelve false

Clase	Método	Descripción
JFrame	Frame()	Constructor de la clase JFrame.
	void setTitle(String titulo)	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.
	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 u evento.
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 txt)	Define una línea de texto que mostrará el componente.

JTextField	JTextField()	Constructor de la clase JTextField.
	String getText()	Retorna el texto contenido en el componente de texto.
JButton	JButton()	Constructor de la clase JButton.
	void setText(String txt)	Define una línea de texto que mostrará el componente.
Event	Object getSource()	El objeto sobre el cual el evento inicialmente ha ocurrido.
ButtonGroup		Crear un conjunto de botones significa que al "encender" uno de esos botones se desactivan todos los demás botones del grupo.
	void clearSelection()	Borra la selección de modo que no se seleccione ninguno de los botones del grupo.
Container		Container es una clase abstracta derivada de Component, que representa a cualquier componente que pueda contener otros componentes.
JPasswordField		Es un componente liviano que permite la edición de una sola línea de texto donde la vista indica que se escribió algo, pero no muestra los caracteres originales.

5. Resultados

Para esta práctica se creo una clase llamada “Principal” en donde por medio de código, se fueron añadiendo componentes para hacer un Login. Se utilizaron componentes como: Jlabel, JTextField, JButton, JPasswordField.

Preguntas.java × Principal.java ×

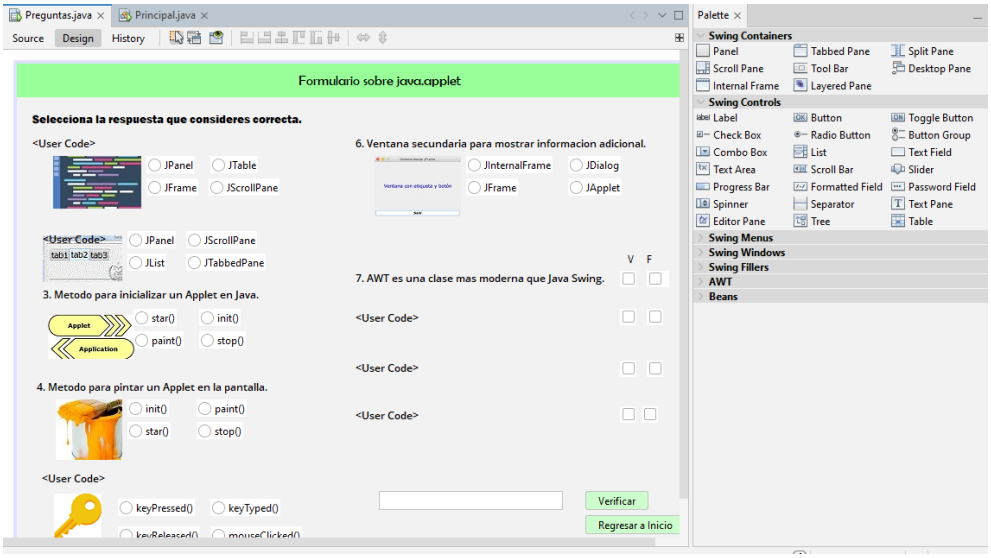
Source History

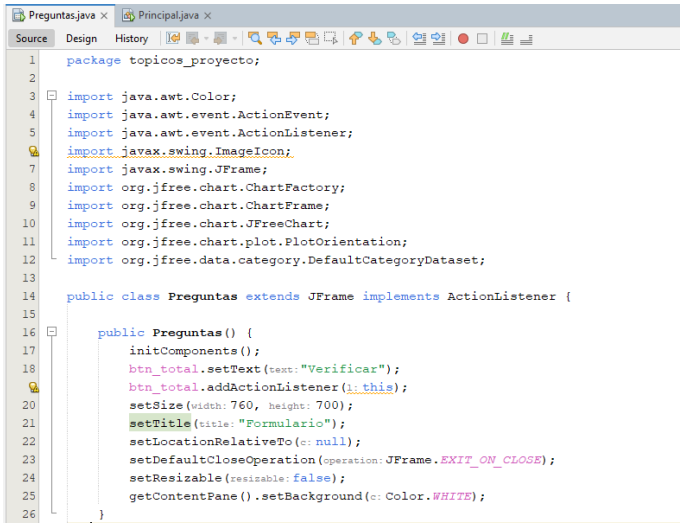
```
package topicos_proyecto;
import java.awt.Color;
import javax.swing.*;
import java.awt.Container;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class Principal extends JFrame implements ActionListener{

    private Container contenedor;
    private JLabel usuario, contrase, Titulo;
    private JTextField usu;
    private JPasswordField contra;
    private JButton IrAFormulario;
    private Preguntas preguntasForm;

    public Principal() {
        initComponents();
        setTitle(title: "Inicio");
        setSize(width: 500, height: 280);
        setLocationRelativeTo(c: null);
        setDefaultCloseOperation(operation: JFrame.EXIT_ON_CLOSE);
        setResizable(resizable: false);
        getContentPane().setBackground(c: Color.WHITE);
    }
    private void initComponents() {
        contenedor = getContentPane();
        contenedor.setLayout(mgr: null);

        Titulo = new JLabel();
        Titulo.setText(text: "Bienvenido");
        Titulo.setBounds(x: 30, y: 30, width: 200, height: 30);
        Font font = new Font(string: "Arial Black", i: Font.BOLD, i1: 16);
        Titulo.setFont(font);

        // Parte del usuario
        usuario = new JLabel();
        usuario.setText(text: "Usuario: ");
        usuario.setBounds(x: 30, y: 80, width: 100, height: 30);
        usu = new JTextField();
        usu.setBounds(x: 130, y: 80, width: 150, height: 30);

        // Parte de la contraseña
        contrase = new JLabel();
        contrase.setText(text: "Contraseña: ");
        contrase.setBounds(x: 30, y: 115, width: 100, height: 30);
        contra = new JPasswordField();
        contra.setBounds(x: 130, y: 115, width: 150, height: 30);

        contenedor.add(comp: Titulo);
        contenedor.add(comp: usuario);
        contenedor.add(comp: usu);
        contenedor.add(comp: contrase);
        contenedor.add(comp: contra);

        IrAFormulario = new JButton(string: "Ir a Formulario");
        IrAFormulario.setBounds(x: 40, y: 170, width: 120, height: 30);
        IrAFormulario.addActionListener(l: this);
        contenedor.add(comp: IrAFormulario);

        ImageIcon imagen = new ImageIcon(string: "D:\\CUARTO SEMESTRE KARYME\\TOPICOS\\UNIDAD_4\\icono.jpg");
        JLabel etiquetaImagen = new JLabel(icon: imagen);
        etiquetaImagen.setBounds(x: 290, y: 20, width: imagen.getIconWidth(),
                height: imagen.getIconHeight());
        contenedor.add(comp: etiquetaImagen);
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == IrAFormulario) {
            String numControlText = usu.getText();
            char[] contraseChars = contra.getPassword();
            String contraseText = new String(chars: contraseChars);

            if (numControlText.equals(anObject: "PROYECTO") && contraseText.equals(anObject: "Topicos")) {
                preguntasForm = new Preguntas();
                preguntasForm.setVisible(b: true);
            } else {
                JOptionPane.showMessageDialog(parentComponent: this,
```

```
                JOptionPane.showMessageDialog(parentComponent: this,
                        message: "Los datos son incorrectos", title: "Error", messageType: JOptionPane.ERROR_MESSAGE);
                }
            }
        }

        public static void main(String[] args) {
                Principal principal = new Principal();
                principal.setVisible(b: true);
        }
    }
```

Para la otra clase, se creo por medio de un JForm para mayor facilidad, ahí solo se fueron arrastrando y acomodando los componentes que nos ofrece Palette.

En la parte del código se ajustó el tamaño del JFrame, su color de fondo y el titulo. También se hizo el código en el botón de “Verificar” para los aciertos correctos e incorrectos y así se pudieran ir contando y almacenando en variables que se ocuparían después a la hora de mostrar la grafica con los resultados.

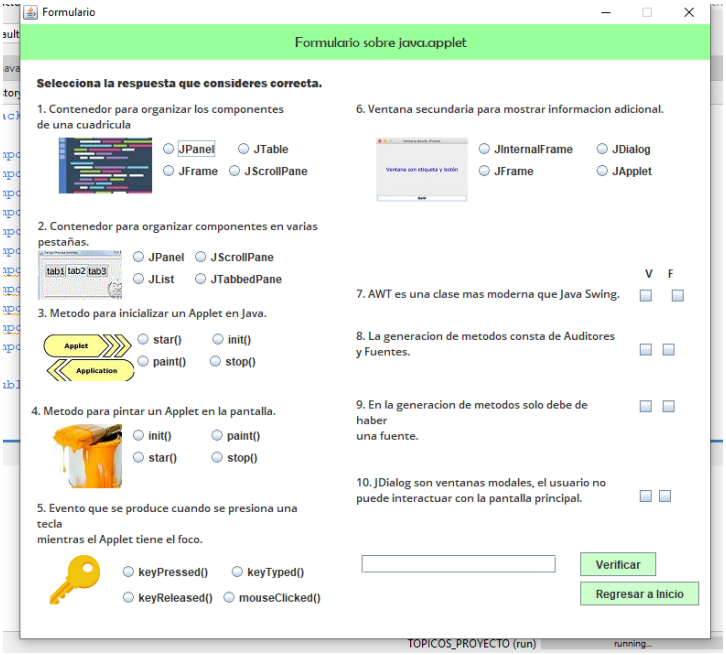


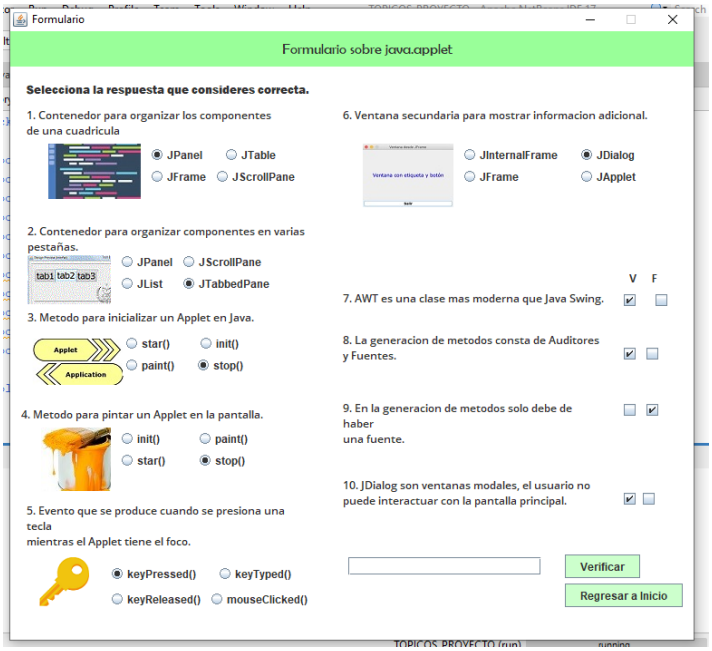


```java
@Override
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == btn_total) {

        int aciertos = 0;
        int errores = 0;

        if (rb_p1r1.isSelected()) {
            aciertos++;
        } else {
            errores++;
        }

        if (rb_p2r2.isSelected()) {
            aciertos++;
        } else {
            errores++;
        }

        if (rb_p3r3.isSelected()) {
            aciertos++;
        } else {
            errores++;
        }
        if (rb_p4r3.isSelected()) {
            aciertos++;
        } else {
            errores++;
        }

        if (rb_p5r1.isSelected()) {
            aciertos++;
        } else {
            errores++;
        }

        if (rb_p6r3.isSelected()) {
            aciertos++;
        } else {
            errores++;
        }

        if (cb_p7r2.isSelected()) {
            aciertos++;
        } else {
            errores++;
        }

        if (cb_p8r1.isSelected()) {
            aciertos++:
        } else {
            errores++;
        }

        if (cb_p9r2.isSelected()) {
            aciertos++;
        } else {
            errores++;
        }

        if (cb_p10r1.isSelected()) {
            aciertos++;
        } else {
            errores++;
        }
        int porcentaje = (100 * aciertos) / 10;

        String mensaje = "Aciertos : " + aciertos + " - " + "Errores : "
                + errores + " - Porc: " + porcentaje + "%";
        tf_total.setText(t: mensaje);

        DefaultCategoryDataset dataset = new DefaultCategoryDataset();
        dataset.setValue(value: aciertos, rowKey: "Resultados", columnKey: "Correctos");
        dataset.setValue(value: errores, rowKey: "Resultados", columnKey: "Incorrectos");
        JFreeChart chart = ChartFactory.createBarChart(title: "A C I E R T O S",
            categoryAxisLabel: "Categoría", valueAxisLabel: "No. de Preguntas", dataset,
            orientation: PlotOrientation.VERTICAL, legend: true, tooltips: true, urls: false);

        chart.getCategoryPlot().getRenderer().setSeriesPaint(i: 0, paint: Color.CYAN);
        chart.getCategoryPlot().getRenderer().setSeriesPaint(i: 1, paint: Color.RED);

        chart.getCategoryPlot().getRangeAxis().setRange(lower: 0, upper: 10);

        ChartFrame frame = new ChartFrame(title: "Gráfica de Barras", chart);
        frame.pack();
        frame.setLocationRelativeTo(c: null);
        frame.setVisible(b: true);

        preg1.clearSelection();
        preg2.clearSelection();
        preg3.clearSelection();
        preg4.clearSelection();
        preg5.clearSelection();
        preg6.clearSelection();
        preg7.clearSelection();
        preg8.clearSelection();
        preg9.clearSelection();
        preg10.clearSelection();

        preg10.clearSelection();

        btn_regresar.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            dispose();
            Principal principal = new Principal();
            principal.setVisible(b: true);
        }
    });
    }
}
```

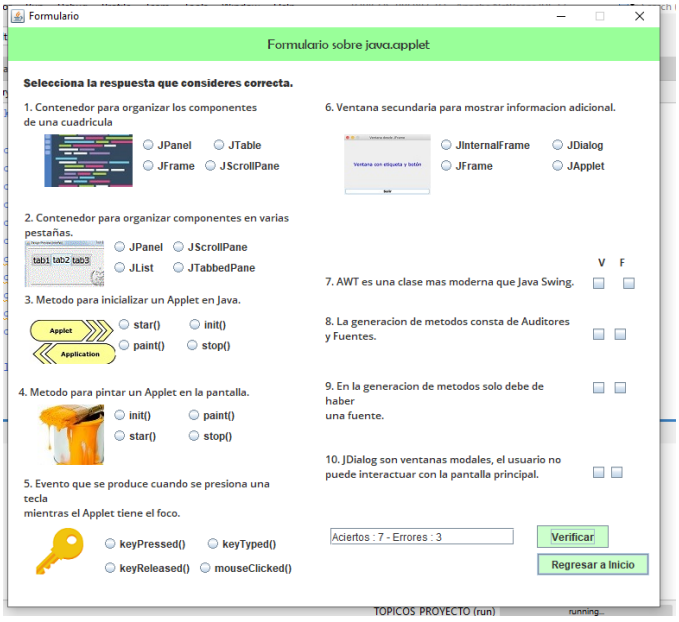
Lo siguiente es el resultado, primero se nos muestra un Login, en donde se tiene que ingresar un usuario y contraseña, si los datos son incorrectos nos muestra una ventana emergente, de lo contrario, nos muestra la siguiente ventana en donde está el formulario con diez preguntas.

Las preguntas están conformadas por JRadioButton y CheckBox en donde solo se puede seleccionar una. Cuando se presiona el botón de “Verificar” nos muestra la grafica con el número de aciertos correctos e incorrectos y también limpia todos los componentes para una nueva oportunidad, también tiene un botón para poder regresar al Login.

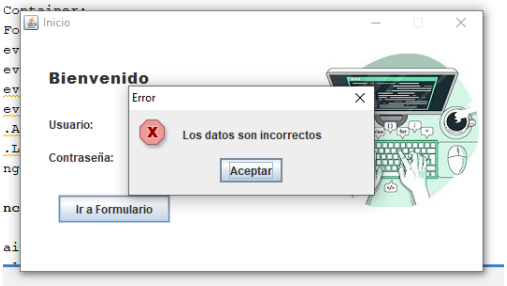



Formulario
Formulario sobre java.applet
Selecciona la respuesta que consideres correcta.
1. Contenedor para organizar los componentes de una cuadricula
JPanel
JTable
JFrame
JScrollPane
2. Contenedor para organizar componentes en varias pestañas.
tab1 tab2 tab3
JPanel
JScrollPane
JList
JTabbedPane
3. Metodo para inicializar un Applet en Java.
Applet
Application
star()
init()
paint()
stop()
4. Metodo para pintar un Applet en la pantalla.
init()
paint()
star()
stop()
5. Evento que se produce cuando se presiona una tecla mientras el Applet tiene el foco.
keyPressed()
keyTyped()
keyReleased()
mouseClicked()
6. Ventana secundaria para mostrar informacion adicional.
Ventana con etiqueta y botón
Aceptar
JInternalFrame
JDialog
JFrame
JApplet
V
F
7. AWT es una clase mas moderna que Java Swing.
8. La generacion de metodos consta de Auditores y Fuentes.
9. En la generacion de metodos solo debe de haber una fuente.
10. JDialog son ventanas modales, el usuario no puede interactuar con la pantalla principal.
Verificar
Regresar a Inicio
TOPICOS_PROYECTO (run)
running...


Formulario sobre java.applet
Selecciona la respuesta que consideres correcta.
Gráfica de Barras
ACIERTOS
No. de Preguntas
Correctos
Incorrectos
Categoria
Resultados
mientras el Applet tiene el foco.
keyPressed()
keyTyped()
keyReleased()
mouseClicked()
Aciertos : 7 - Errores : 3
Verificar
Regresar a Inicio


Formulario
Formulario sobre java.applet
Selecciona la respuesta que consideres correcta.
Aciertos : 7 - Errores : 3
Verificar
Regresar a Inicio
TOPICOS_PROYECTO (run)
running...


Inicio
Bienvenido
Usuario:
TRABAJO
Contraseña:
Ir a Formulario


Inicio
Bienvenido
Usuario:
Contraseña:
Error
Los datos son incorrectos
Aceptar
Ir a Formulario

6. Conclusiones

En conclusión el uso de JFrame y sus componentes en la creación de interfaces gráficas de usuario nos ofrecen la capacidad de construir aplicaciones más sofisticadas, interactivas y visualmente atractivas. Nos abren nuevas posibilidades y permiten que las aplicaciones se destaquen al brindar una experiencia más intuitiva y agradable al usuario. La selección adecuada de componentes, su disposición en la ventana y su interacción con el usuario son aspectos cruciales para crear interfaces eficientes y funcionales.

También debemos tener en cuenta aspecto importantes como la optimización del rendimiento, la gestión de eventos, la validación de la entrada del usuario y la accesibilidad. Estos elementos contribuyen a la creación de aplicaciones de alta calidad que cumplen con los estándares actuales y brindan una experiencia óptima para los usuarios.

Cuando dominamos todos estos conceptos, podemos mejorar la interacción con los usuarios, crear experiencias más agradables y construir aplicaciones más completas y versátiles en el mundo de la programación actual.