Universidad Mariano Galvez

Proyecto Final Programacion

Manual de Usuario

Este Manual de usuario es para que todo el mundo pueda usar el programa con facilidad y no existan problemas al interactuar con el. Se explicara lo que hace el codigo y que funciones cuenta.

Pasos iniciales

- Deberia de tener instalado Apache NetBeans 21 si no lo tienes instalado aqui esta el link de descarga: https://netbeans.apache.org/front/main/download/nb21/
- Ya instalado el programa deberias de poder abrir nuestro proyecto en https://github.com/
- Ya abierto todo esto el proyecto deberia de poder funcionar si lo ejecutas.

Programa similar a Excel

El programa si se dan cuenta tiene varia opciones de uso Como sumar y multiplicar como agregar nuevas hojas con un nombre que deseas. Este proyecto esta realizado con las siguientes clases.

- Controlador (Main)
- Vista
- Modelo

Controlador (Main)

En esta clase contamos con la funcion main donde se tendra que correr el programa igualmente cuenta con otras funciones de sumar, multiplicar y Hojas nuevas.

Codigo

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class Controlador {
private Modelo modelo;
private VISTA VISTA;
```

```
public Controlador(Modelo modelo, VISTA vista) {
    this.modelo = modelo;
    this.VISTA = vista;

    vista.agregarSumarListener(new SumarListener());
    vista.agregarMultiplicarListener(new MultiplicarListener());
    vista.agregarHojaListener(new AgregarHojaListener());
    vista.agregarSelectorHojasListener(new SelectorHojasListener());

    modelo.agregarHoja("Hoja1");
    vista.agregarHojaAlSelector("Hoja1");
}
```

```
class SumarListener implements ActionListener {
    @Override
    public void actionPerformed(ActionEvent e) {
        int fila1 = VISTA.getFila1();
        int columna1 = VISTA.getColumna1();
        int fila2 = VISTA.getFila2();
        int columna2 = VISTA.getColumna2();
        int resultado = modelo.getHojaActual().sumarCeldas(fila1, columna1, fila2, columna2);
        VISTA.setResultado(resultado);
    }
}
```

```
class MultiplicarListener implements ActionListener {
    @Override
    public void actionPerformed(ActionEvent e) {
        int fila1 = VISTA.getFila1();
        int columna1 = VISTA.getColumna1();
        int fila2 = VISTA.getFila2();
        int columna2 = VISTA.getColumna2();
        int resultado = modelo.getHojaActual().multiplicarCeldas(fila1, columna1, fila2, columna2);
        VISTA.setResultado(resultado);
    }
}

class AgregarHojaListener implements ActionListener {
    @Override
    public void actionPerformed(ActionEvent e) {
        String nombreNuevaHoja = VISTA.getNombreNuevaHoja();
        modelo.agregarHoja(nombreNuevaHoja);
        VISTA.agregarHojaAlSelector(nombreNuevaHoja);
    }
}
```

```
class SelectorHojasListener implements ActionListener {
    @Override
    public void actionPerformed(ActionEvent e) {
        String hojaSeleccionada = VISTA.getHojaSeleccionada();
        modelo.cambiarHoja(hojaSeleccionada);
    }
}
```

```
public static void main(String[] args) {
Modelo modelo = new Modelo();
VISTA vista = new VISTA();
Controlador controlador = new Controlador(modelo,
vista);
vista.setVisible(true);
}

}
```

Vista

Esta clase crea una interfaz gráfica para una hoja de cálculo básica que permite al usuario ingresar valores en celdas, realizar operaciones como suma y multiplicación en filas y columnas específicas, y gestionar múltiples hojas de cálculo.

Codigo

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionListener;
```

```
public class VISTA extends JFrame {
private JTextField[][] camposCeldas;
private JButton botonSumar, botonMultiplicar;
private JTextField campoFila1, campoColumna1,
campoFila2, campoColumna2;
private JLabel etiquetaResultado;
private JComboBox<String> selectorHojas;
private JButton botonAgregarHoja;
private JTextField campoNombreNuevaHoja;
```

```
public VISTA() {
    setTitle("Hoja de Cálculo");
    setSize(600, 400);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setLayout(new BorderLayout());
```

```
JPanel panelHoja = new JPanel(new GridLayout(11, 10));
camposCeldas = new JTextField[10][10];

// Etiquetas de las columnas (letras)
panelHoja.add(new JLabel());
for (char c = 'A'; c <= 'J'; c++) {
    panelHoja.add(new JLabel(String.valueOf(c)));
}
```

```
    for (int i = 0; i < 10; i++) {
        // Etiquetas de las filas (números)
        panelHoja.add(new JLabel(String.valueOf(i + 1)));

        for (int j = 0; j < 10; j++) {
            camposCeldas[i][j] = new JTextField();
            panelHoja.add(camposCeldas[i][j]);
        }
    }

    JPanel panelControl = new JPanel();
    panelControl.setLayout(new GridLayout(3, 2));
```

```
campoFila1 = new JTextField();
campoColumna1 = new JTextField();
campoFila2 = new JTextField();
campoColumna2 = new JTextField();
botonSumar = new JButton("Sumar");
botonMultiplicar = new JButton("Multiplicar");
etiquetaResultado = new JLabel("Resultado: ");

panelControl.add(new JLabel("Fila 1:"));
panelControl.add(campoFila1);
panelControl.add(new JLabel("Columna 1:"));
panelControl.add(campoColumna1);
panelControl.add(new JLabel("Fila 2:"));
panelControl.add(campoFila2);
panelControl.add(new JLabel("Columna 2:"));
panelControl.add(campoColumna2);
panelControl.add(botonSumar);
panelControl.add(botonMultiplicar);
panelControl.add(etiquetaResultado);
```

```
    JPanel panelControlHojas = new JPanel();
    campoNombreNuevaHoja = new JTextField(10);
    botonAgregarHoja = new JButton("Agregar Hoja");
    selectorHojas = new JComboBox<>();

    panelControlHojas.add(new JLabel("Nueva Hoja:"));
    panelControlHojas.add(campoNombreNuevaHoja);
    panelControlHojas.add(botonAgregarHoja);
    panelControlHojas.add(selectorHojas);

    add(panelHoja, BorderLayout.CENTER);
    add(panelControl, BorderLayout.SOUTH);
    add(panelControlHojas, BorderLayout.NORTH);
}
```

```
public void setValorCelda(int fila, int columna, int valor) {
    camposCeldas[fila][columna].setText(String.valueOf(valor));
}

public int getValorCelda(int fila, int columna) {
    return Integer.parseInt(camposCeldas[fila][columna].getText());
}

public int getFila1() {
    return Integer.parseInt(campoFila1.getText());
}

public int getColumna1() {
    return Integer.parseInt(campoColumna1.getText());
}

public int getFila2() {
    return Integer.parseInt(campoFila2.getText());
}
```

```
public int getColumna2() {
    return Integer.parseInt(campoColumna2.getText());
}

public void setResultado(int resultado) {
    etiquetaResultado.setText("Resultado: " + resultado);
}

public void agregarSumarListener(ActionListener listener) {
    botonSumar.addActionListener(listener);
}

public void agregarMultiplicarListener(ActionListener listener) {
    botonMultiplicar.addActionListener(listener);
}

public void agregarHojaListener(ActionListener listener) {
    botonAgregarHoja.addActionListener(listener);
}
```

```
public void agregarSelectorHojasListener(ActionListener listener) {
    selectorHojas.addActionListener(listener);
}

public void agregarHojaAlSelector(String nombreHoja) {
    selectorHojas.addItem(nombreHoja);
}

public String getHojaSeleccionada() {
    return (String) selectorHojas.getSelectedItem();
}

public String getNombreNuevaHoja() {
    return campoNombreNuevaHoja.getText();
}
```

}

Modelo

Este código establece una estructura básica para una aplicación de hoja de cálculo en Java, con clases para el modelo de datos, las hojas de cálculo y las celdas.

Codigo

```
import java.util.HashMap;

public class Modelo {
private HashMap<String, Hoja> hojas;
private Hoja hojaActual;
```

```
public Modelo() {
    hojas = new HashMap<>();
    agregarHoja("Hoja1"); // Agregar una hoja por defecto al crear el modelo
}

// Método para agregar una nueva hoja
public void agregarHoja(String nombre) {
    hojas.put(nombre, new Hoja(nombre));
    hojaActual = hojas.get(nombre);
}
```

```
// Método para cambiar a una hoja existente
public void cambiarHoja(String nombre) {
    if (hojas.containsKey(nombre)) {
        hojaActual = hojas.get(nombre);
    }
}

// Método para eliminar una hoja existente
public void eliminarHoja(String nombre) {
    if (hojas.containsKey(nombre)) {
        hojas.remove(nombre);
    }
}

// Método para obtener todas las hojas
public HashMap<String, Hoja> getHojas() {
    return hojas;
}
```

```
// Método para obtener la hoja actual
public Hoja getHojaActual() {
    return hojaActual;
}
```

}

```
class Hoja {
private String nombre;
private Celda[][] celdas;
private static final int TAMANO = 10; // Tamaño de la hoja 10x10
```

```
public Hoja(String nombre) {
    this.nombre = nombre;
    celdas = new Celda[TAMANO][TAMANO];
    for (int i = 0; i < TAMANO; i++) {
        for (int j = 0; j < TAMANO; j++) {
            celdas[i][j] = new Celda();
        }
    }
}

public String getNombre() {
    return nombre;
}
```

```
public void setValorCelda(int fila, int columna, int valor) {
    celdas[fila][columna].setValor(valor);
}

public int getValorCelda(int fila, int columna) {
    return celdas[fila][columna].getValor();
}

public int sumarCeldas(int fila1, int columna1, int fila2, int columna2) {
    return celdas[fila1][columna1].getValor() + celdas[fila2][columna2].getValor();
}

public int multiplicarCeldas(int fila1, int columna1, int fila2, int columna2) {
    return celdas[fila1][columna1].getValor() * celdas[fila2][columna2].getValor();
}
```

}

class Celda {

private int valor;

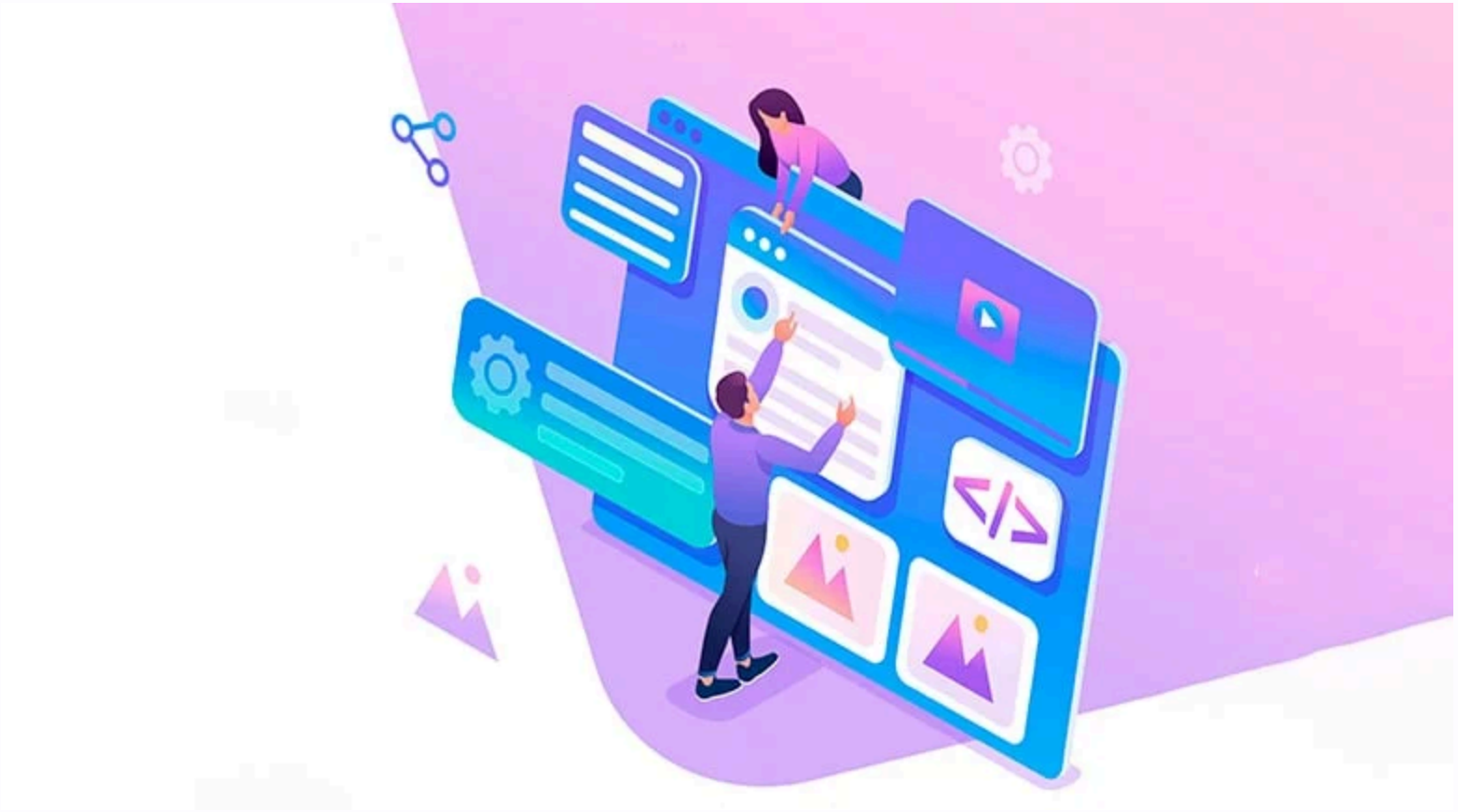
```
public Celda() {
    valor = 0;
}

public int getValor() {
    return valor;
}

public void setValor(int valor) {
    this.valor = valor;
}
```

}

Interfaz del usuario



Podras ingresar datos a cada celda que podras sumar y multiplicar igualmente si miras en la parte superior derecha indica agregar nuevas hojas para contar con varias a la vez y poder usarlas.

Este proyecto si te metes al github te daras cuenta que esta realizado con tablas hash que lo que realiza es hacer la tablas y celdas que contenga un almacenador de datos por eso cuando seleccionas una celda no selecciona toda la fila o columna si no que solo esa celda y puedes almacenar numeros como letras.

Tendras unos botones en la parte inferior sonde podras sumar y multiplicar los valores de la celdas en filas y columnas y tendras el resultado en la parte inferior derecho.