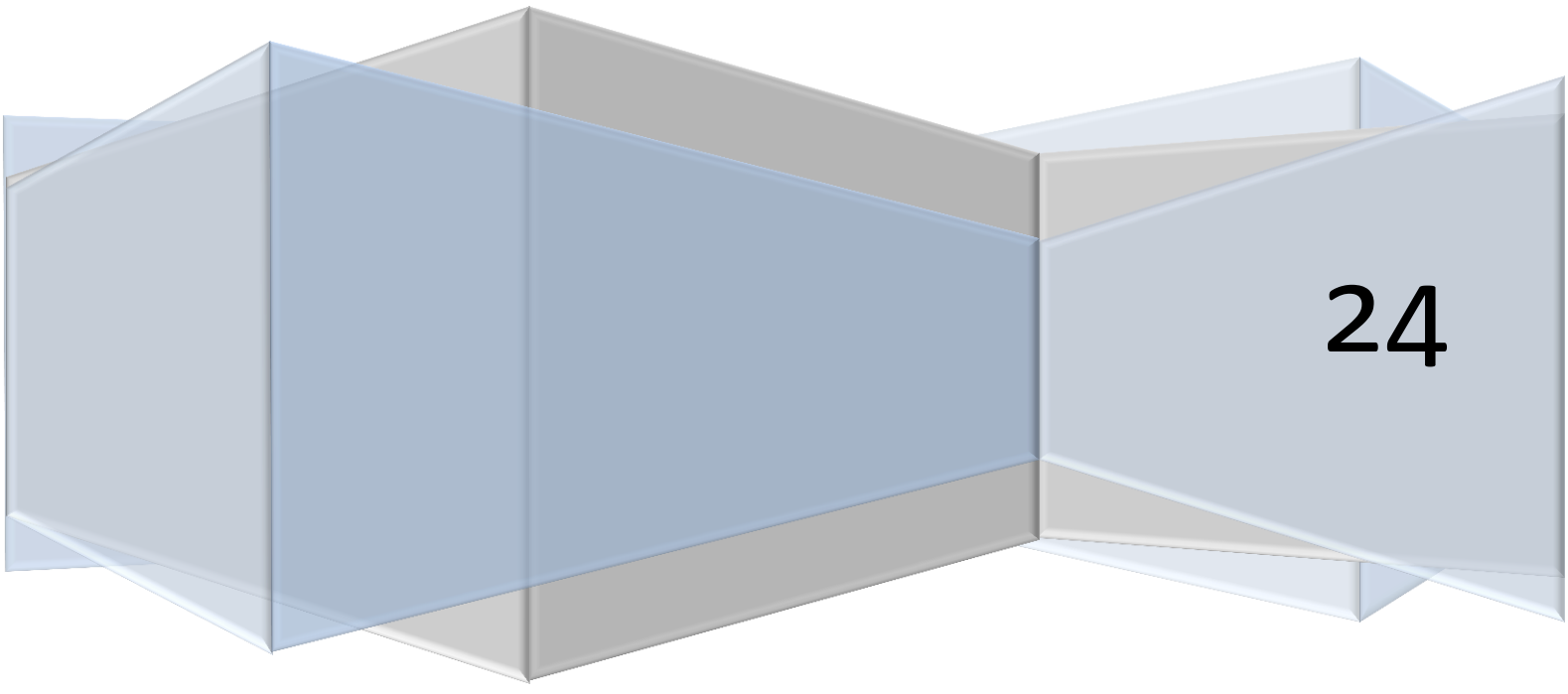
Tarea Final Tema 6

Programación

Héctor Mora Sánchez



Índice

1. Importaciones de paquetes:

javax.swing.*: Importa todas las clases del paquete Swing, que se utiliza para crear interfaces gráficas de usuario (GUI) en Java.

java.awt.*: Importa todas las clases del paquete AWT, que proporciona clases para crear interfaces gráficas de usuario, manejar eventos, etc.

java.awt.event.*: Importa clases para manejar eventos de usuario.

java.io.*: Importa clases para manejar operaciones de entrada y salida, como leer y escribir archivos.

```
3   import javax.swing.*;
4   import java.awt.*;
5   import java.awt.event.*;
6   import java.io.*;
```

2. Declaración de la clase TextEditor:

La clase **EditorTexto** extiende **JFrame**, lo que significa que es una ventana de aplicación.

Implementa la interfaz **ActionListener** para manejar eventos de acción, como hacer clic en un botón del menú.

```
public class EditorTexto extends JFrame implements ActionListener {
```

3. Declaración de variables:

JTextArea textArea: Área de texto donde se muestra y edita el contenido del archivo.

JFileChooser fileChooser: Selector de archivos para abrir y guardar archivos.

```
    JTextArea textArea;  4 usages
    JFileChooser fileChooser;  5 usages
```

4. Constructor EditorTexto:

Configura el **título** y el **tamaño** de la ventana.

Configura la **operación de cierre** al hacer clic en el botón de cierre de la ventana para que la aplicación se cierre correctamente.

Crea un **JTextArea** dentro de un **JScrollPane** y lo agrega al centro de la ventana.

Crea un **JMenuBar** con un menú "**File**" y dos elementos de menú "**Open**" y "**Save**", y los agrega a la barra de menú.

Asigna escuchadores de eventos a los elementos del menú "**Open**" y "**Save**".

```
12      public EditorTexto() {  1 usage
13          setTitle("Simple Text Editor");
14          setSize( width: 600,  height: 400);
15          setDefaultCloseOperation(EXIT_ON_CLOSE);
16
17          textArea = new JTextArea();
18          JScrollPane scrollPane = new JScrollPane(textArea);
19          add(scrollPane, BorderLayout.CENTER);
20
21          JMenuBar menuBar = new JMenuBar();
22          JMenu fileMenu = new JMenu( s: "File");
23          JMenuItem openItem = new JMenuItem( text: "Open");
24          JMenuItem saveItem = new JMenuItem( text: "Save");
25          openItem.addActionListener( l: this);
26          saveItem.addActionListener( l: this);
27          fileMenu.add(openItem);
28          fileMenu.add(saveItem);
29          menuBar.add(fileMenu);
30          setJMenuBar(menuBar);
31
32          fileChooser = new JFileChooser();
33      }
```

5. Método ActionPerformed:

Este método maneja los **eventos de acción** generados por los elementos del menú.

Si se hace clic en "**Open**", muestra un **cuadro de diálogo** para seleccionar un archivo de texto, lee su contenido y lo muestra en el **JTextArea**.

```
35      public void actionPerformed(ActionEvent e) {
36          if (e.getActionCommand().equals("Open")) {
37              int returnValue = fileChooser.showOpenDialog( parent: this);
38              if (returnValue == JFileChooser.APPROVE_OPTION) {
39                  File selectedFile = fileChooser.getSelectedFile();
40                  try {
41                      FileReader fileReader = new FileReader(selectedFile);
42                      BufferedReader reader = new BufferedReader(fileReader);
43                      String line;
44                      StringBuilder stringBuilder = new StringBuilder();
45                      while ((line = reader.readLine()) != null) {
46                          stringBuilder.append(line).append("\n");
47                      }
48                      reader.close();
49                      textArea.setText(stringBuilder.toString());
50                  } catch (IOException ex) {
51                      ex.printStackTrace();
52                  }
53              }
```

Si se hace clic en "**Save**", muestra un cuadro de diálogo para seleccionar la ubicación y el nombre del archivo, y guarda el contenido del **JTextArea** en el archivo seleccionado.

```
54          } else if (e.getActionCommand().equals("Save")) {
55              int returnValue = fileChooser.showSaveDialog( parent: this);
56              if (returnValue == JFileChooser.APPROVE_OPTION) {
57                  File selectedFile = fileChooser.getSelectedFile();
58                  try {
59                      FileWriter fileWriter = new FileWriter(selectedFile);
60                      fileWriter.write(textArea.getText());
61                      fileWriter.close();
62                  } catch (IOException ex) {
63                      ex.printStackTrace();
64                  }
65              }
66          }
67      }
68
```

6. Método main:

Este método es el punto de entrada principal del programa. Crea una instancia de **EditorTexto** y la hace visible en la interfaz gráfica.

```
68
69      public static void main(String[] args) {
70          EditorTexto editor = new EditorTexto();
71          editor.setVisible(true);
72      }
73  }
74
```