

TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE TLAXIACO

INGENIERÍA EN SISTEMAS COMPUTACIONALES
DESARROLLO DE SOFTWARE |SIE-DES-2022-01

SCD – 1003 ARQUITECTURA DE COMPUTADORAS

CÁTEDRA DEL ING. OSORIO SALINAS EDWARD

ALUMNO:

No	Nombre	No de Control
01	Velasco López Daniel	22620076

GRUPO:

5BS

CAMBIO DE BASE

REPORTE DE PRÁCTICA INDIVIDUAL

TEMA 1: ARQUITECTURA DE COMPUTADORAS

Tlaxiaco, Oaxaca. A 02 de sep. de 24



Boulevard Tecnológico Km. 2.5, Llano Yosovee C.P. 69800. Tlaxiaco, Oaxaca. Tel. (953) 55 21322 y (953) 55 20405, e-mail: dir_tlaxiaco@tecnm.mx; tecnm.mx | tlaxiaco.tecnm.mx



Contenido.

Descripción.....	4
Programa Cambio de Base.....	5
Descripción detallada del programa Cambio de Base.....	7
Paquete y Bibliotecas importadas.....	7
Definición de Clase y Componentes	8
Constructor de la clase	9
Configuración del Panel de Entrada	10
Configuración del Panel de Base de Entrada.....	11
Configuración del Panel de Base de Salida	12
Configuración del Panel de Botones.....	13
Configuración del Área de Salida	14
Componentes a las ventanas	14
Manejo de Eventos.....	15
Método GetBase	16
Método Main	17
Resultado.....	18
Conclusión	21

Lista de Figuras.

Ilustración 1 Diagrama de Flujo.....	6
Ilustración 2 Paquete de Clase	7
Ilustración 3 Bibliotecas	7
Ilustración 4 Clase: Cambio_de_Base.....	8
Ilustración 5 Componentes de Interfaz	8
Ilustración 6 Constructor de Clase	9
Ilustración 7 Configuración de ventana principal.....	9
Ilustración 8 Panel de Entrada	10
Ilustración 9 Panel de Base de Entrada.....	11
Ilustración 10 Panel de Base de Salida	12
Ilustración 11 Panel de Botones.....	13
Ilustración 12 Botón de Conversión	13
Ilustración 13 Botón de Limpiar.....	13
Ilustración 14 Área de salida	14
Ilustración 15 Nuevos paneles al marco.....	14
Ilustración 16 Eventos	15
Ilustración 17 Método getBase.....	16
Ilustración 18 Método main	17
Ilustración 19 Panel Principal.....	18
Ilustración 20 Número que deseamos convertir	18
Ilustración 21 Número a base de entrada.....	19
Ilustración 22 Número a base de salida	19
Ilustración 23 Resultado de conversión.....	20
Ilustración 24 Botón Limpiar	20

Descripción

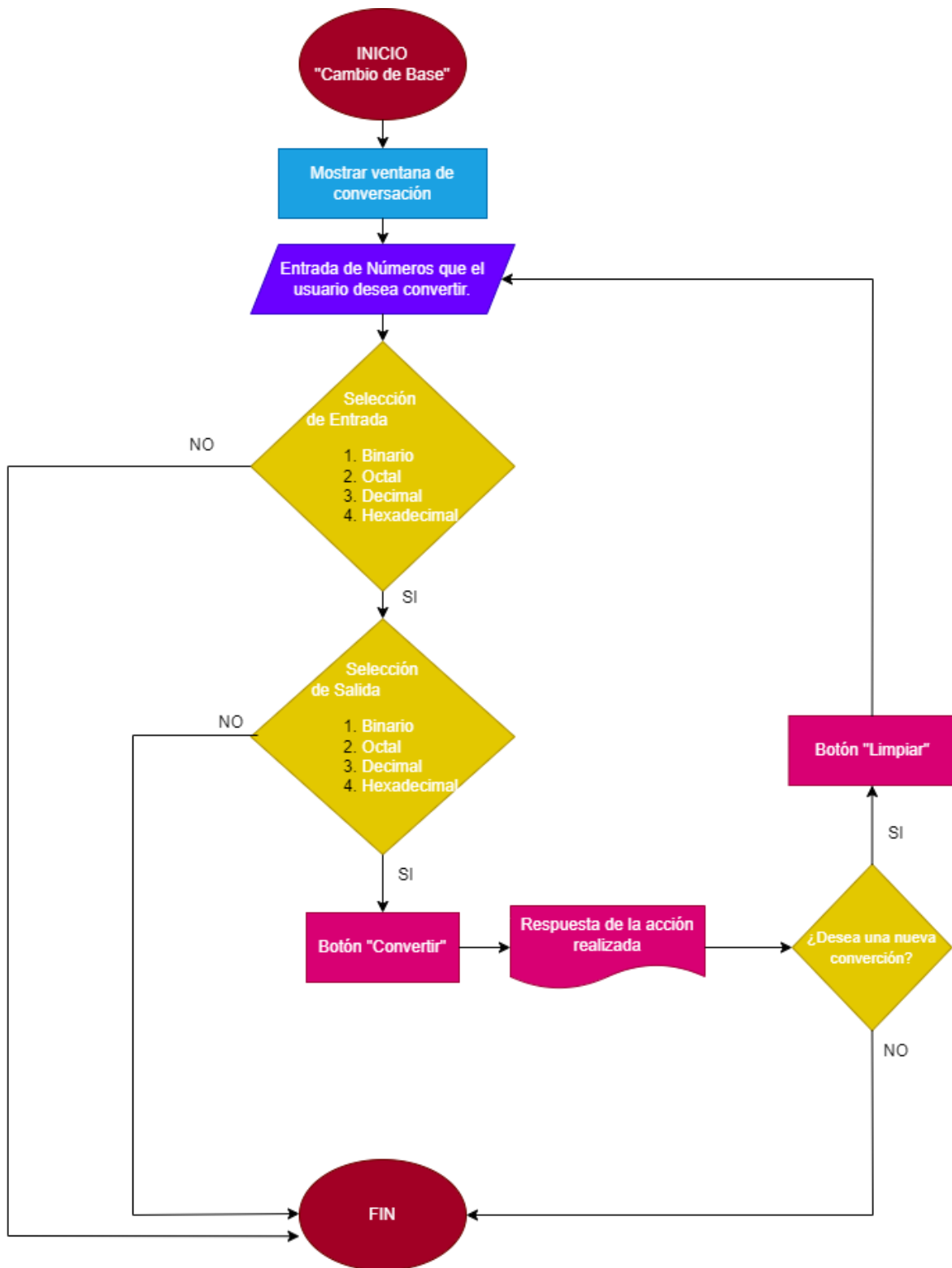
El diseño de programas informáticos eficaces y útiles comienza con una planificación cuidadosa, y una de las herramientas más valiosas en esta etapa es el diagrama de flujo. Este documento describe la implementación de un programa para convertir números entre diferentes sistemas de numeración, un importante proceso computacional y de ingeniería. La representación gráfica de los pasos del proceso facilita la comprensión de la lógica subyacente, predice posibles errores y optimiza el tiempo de desarrollo. Los programas resultantes no sólo simplifican la tarea de convertir números entre bases como binaria, octal, decimal y hexadecimal, sino que también proporcionan una interfaz gráfica intuitiva que mejora la experiencia del usuario.

Programa Cambio de Base

Antes de desarrollar un programa, es fundamental crear un diagrama de flujo, ya que este sirve como una representación visual del proceso que debe seguir el programa. Un diagrama de flujo no solo ayuda a planificar y visualizar la lógica antes de escribir el código, sino que también ofrece varios beneficios, como la claridad en la estructura de lo que se planea implementar, la identificación temprana de errores, y un aspecto crucial en la programación: la optimización del tiempo.

El siguiente diagrama ilustra el desarrollo de un programa diseñado para convertir números entre diferentes sistemas numéricos. Los usuarios pueden ingresar números en una base específica (como binaria, octal, decimal o hexadecimal) en el área de texto. El programa procesa el valor secuencialmente, lo convierte a la base requerida y muestra el resultado con precisión. Esta funcionalidad es esencial en diversos campos de la informática y la ingeniería, donde convertir bases numéricas es una tarea común pero esencial. El programa está diseñado para simplificar y automatizar este proceso para que los usuarios puedan realizar conversiones de forma rápida y eficiente.

Ilustración 1 Diagrama de Flujo



Descripción detallada del programa Cambio de Base

“Cambio de Base”, es una aplicación de escritorio desarrollada con lenguaje Java que permite convertir números de una base numérica a otra. El usuario puede ingresar un número en una base específica(binario, octal, decimal o hexagonal) y convertirlo a otra base de su selección

Paquete y Bibliotecas importadas

Continuando con la elaboración del programa, empezamos por la creación de un paquete. Esto ayuda a organizar las clases en grupos lógicos y evita conflictos de nombres. En este caso, es “com.mycompany.cambio_de_clase”.

Ilustración 2 Paquete de Clase

```
1 package com.mycompany.cambio_de_base;  
2
```

Se importan las bibliotecas necesarias para la interfaz gráfica y así poder manejar los eventos que hacen funcionar a nuestro programa.

Ilustración 3 Bibliotecas

```
3 import javax.swing.*;  
4 import java.awt.*;  
5 import java.awt.event.ActionEvent;  
6 import java.awt.event.ActionListener;  
7 import javax.swing.border.EmptyBorder;  
8 import javax.swing.border.LineBorder;
```

- import javax.swing.*; Swing es un conjunto de componentes gráficos para construir interfaces de usuario en aplicaciones java. Se utiliza para crear ventanas, paneles y otros elementos gráficos.
- import java.awt.*; proporciona clases para el diseño y la gestión de la interfaz gráfica del usuario. Se utiliza para configurar colores, tamaños y disposición de los componentes gráficos.
- import java.awt.event.ActionEvent; se utiliza en la implementación de manejadores de eventos para capturar y responder a eventos generados por el usuario, como presionar un botón.

- `import java.awt.event.ActionListener;` esta interfaz se utiliza para definir el comportamiento que debe ocurrir cuando se produce un evento de acción.
- `import javax.swing.border.EmptyBorder;` esta clase se utiliza para proporcionar un margen interno(espacio) alrededor de los componentes gráficos, para mejorar la presentación visual.
- `import javax.swing.border.LineBorder;` al igual que `EmptyBoder`, se utiliza para agregar un borde visual alrededor de los componentes gráficos, lo que puede ayudar a resaltar o separar visualmente los componentes en la interfaz de usuario.

Definición de Clase y Componentes

Definir la clase “Cambio_de_Base” extendido de “JFrame” e implementando “ActionListener”, combina la funcionalidad de creación de ventanas con la capacidad de manejar eventos de usuario, proporcionando una base sólida para construir una interfaz gráfica interactiva en la aplicación.

Ilustración 4 Clase: Cambio_de_Base

```
10 public class Cambio_de_Base extends JFrame implements ActionListener{
```

Las líneas de código que definen los componentes gráficos son esenciales para crear una interfaz de usuario en el programa. Cada uno de estos componentes tiene un papel específico en la funcionalidad de lo que pretendemos desarrollar.

Ilustración 5 Componentes de Interfaz

```
11 private final JTextField inputField;  
12 private final JComboBox<String> baseInput;  
13 private final JComboBox<String> baseOutput;  
14 private final JTextArea outputArea;  
15 private final JButton convertButton;  
16 private final JButton clearButton;
```

- `private final JTextField inputField;` declara un campo de texto (`JTextField`) llamado (`inputField`) lo que les permite a los usuarios ingresar un número que será convertido entre diferentes bases.

Cambio de Base

- `private final JComboBox<String> baseInput;` declara un menú desplegable ofreciendo a los usuarios seleccionar la base en la que está el número ingresado para que el programa pueda interpretarlo correctamente durante la conversión.
- `private final JComboBox<String> baseOutput;` declara un segundo menú desplegable permitiendo a los usuarios elegir la base a la que desean convertir el número ingresado con anterioridad, lo cual define cómo se formateará el resultado.
- `private final JTextArea outputArea;` declara un área de texto donde se muestra el resultado de la conversión o mensajes de error.
- `private final JButton convertButton;` el botón llama al método que realiza la conversión de bases y actualiza "outputArea" con el resultado.
- `private final JButton clearButton;` permite a los usuarios limpiar los campos de entrada y salida, permitiendo al usuario iniciar una nueva conversión sin tener que hacerlo manualmente.

Constructor de la clase

Nuestro constructor de la clase "Cambio_de_Base" inicializa y configura la interfaz gráfica cuando se crea la instancia (son la materialización de una clase, es decir, son objetos concretos que tienen su propio estado) de esta clase.

Ilustración 6 Constructor de Clase

```
18 public Cambio_de_Base() {
```

Las configuraciones iniciales son cruciales para asegurar que la ventana de la aplicación se muestre de manera adecuada y funcional, proporcionando una experiencia de usuario consistente y efectiva.

Ilustración 7 Configuración de ventana principal

```
19 setTitle( title: "Conversor de Bases");
20 setSize( width: 400, height: 300);
21 setDefaultCloseOperation( operation: JFrame.EXIT_ON_CLOSE);
22 setLayout( new GridLayout( rows: 6, cols: 1)); // Aumentado para acomodar el botón de limpiar
```

- `setTitle` define el título de la ventana.
- `setSize` establece el tamaño de nuestra ventana principal.
- `setDefaultCloseOperation` asegura que la aplicación se cierre al cerrar la ventana.

- `setLayout(new GridLayout(6, 1))` utiliza un `GridLayout`(organiza los componentes de un contenedor en una cuadrícula rectangular) con 6 filas y 1 columna para organizar los componentes en la ventana.

Configuración del Panel de Entrada

Ilustración 8 Panel de Entrada

```
24 // Configurar el color de fondo y márgenes del panel de entrada
25 JPanel inputPanel = new JPanel();
26 inputPanel.setBackground(new Color( r:135, g:206, b:235)); // Color azul claro
27 inputPanel.setBorder( border:BorderFactory.createCompoundBorder(
28     new LineBorder( color:Color.BLUE, thickness:5), // Borde azul de 5 píxeles
29     new EmptyBorder( top:10, left:10, bottom:10, right:10) // Margen interno
30 ));
31 inputPanel.add(new JLabel( text:"Número:"));
32 inputField = new JTextField( columns:15);
33 inputPanel.add( comp:inputField);
```

- `JPanel inputPanel = new JPanel();` se usa para contener otros componentes gráficos como etiquetas, campos de texto, y facilitar su organización en la interfaz de usuario.
- `inputPanel.setBackground(new Color(135, 206, 235));` // Color azul claro, define un color específico (en este caso, un azul claro) para el fondo del panel utilizando la clase `Color` con valores RGB (especifican la intensidad de los colores rojo, verde y azul para mostrar un color específico).
- `inputPanel.setBorder(BorderFactory.createCompoundBorder(new LineBorder(Color.BLUE, 5), // Borde azul de 5 píxeles new EmptyBorder(10, 10, 10, 10) // Margen interno));` define dos tipos de bordes para el panel, `LineBorder(Color.BLUE, 5)`: Un borde sólido de 5 píxeles de grosor con color azul y `EmptyBorder(10, 10, 10, 10)`: Un margen interno de 10 píxeles en todos los lados del panel, creando espacio entre el borde y el contenido del panel.
- `inputPanel.add(new JLabel("Número:"))`; se utiliza para mostrar un texto estático en la interfaz de usuario, en este caso, el `JLabel` muestra el texto "Número:" para identificar al usuario qué información debe de ingresar en el campo de texto.
- `inputField = new JTextField(15);` se especifica un tamaño de 15 columnas para el campo de texto, que determina su ancho. `inputField` se utilizará para que los usuarios ingresen el número que desean convertir.

Cambio de Base

- `inputPanel.add(inputField);` permite que el campo de texto se muestre debajo de la etiqueta "Número:" en el panel, de modo que los usuarios puedan ingresar el número que desean convertir.

Configuración del Panel de Base de Entrada

Ilustración 9 Panel de Base de Entrada

```
35 // Configurar el color de fondo y márgenes del panel de base de conversión
36 JPanel baseInputPanel = new JPanel();
37 baseInputPanel.setBackground(new Color(r:173, g:216, b:230)); // Color azul cielo
38 baseInputPanel.setBorder(BorderFactory.createCompoundBorder(
39     new LineBorder(Color.BLUE, thickness:5), // Borde azul de 5 píxeles
40     new EmptyBorder(top:10, left:10, bottom:10, right:10) // Margen interno
41 ));
42 baseInputPanel.add(new JLabel(text:"Base de entrada:"));
43 String[] bases = {"Binario", "Octal", "Decimal", "Hexadecimal"};
44 baseInput = new JComboBox<>(items:bases);
45 baseInputPanel.add(comp:baseInput);
```

Este bloque de código organiza y configura la parte de la interfaz de usuario donde el usuario puede elegir la base del número a convertir. La estructura del panel y los componentes garantizan que la selección de la base de entrada sea intuitiva y visualmente agradable.

- `JPanel baseInputPanel = new JPanel();` es el panel que contendrá el `JLabel` y el `JComboBox` para que el usuario seleccione la base numérica de entrada.
- `baseInputPanel.setBackground(new Color(173, 216, 230));` // Color azul cielo, ayuda a destacar este panel dentro de la interfaz, diferenciándolo de otros paneles y facilitando la navegación visual para el usuario.
- `baseInputPanel.setBorder(BorderFactory.createCompoundBorder(`
 `new LineBorder(Color.BLUE, 5),` // Borde azul de 5 píxeles
 `new EmptyBorder(10, 10, 10, 10)` // Margen interno
); configura el borde del panel, utilizando un borde compuesto, el borde azul resalta visualmente el panel, mientras que el margen interno asegura que el contenido (etiqueta y caja de selección) no esté demasiado pegado al borde, mejorando la legibilidad y apariencia.
- `baseInputPanel.add(new JLabel("Base de entrada:"));` la etiqueta "Base de entrada:" informa al usuario que debe seleccionar la base numérica del número que desean convertir.

- `String[] bases = {"Binario", "Octal", "Decimal", "Hexadecimal"};` este arreglo contiene las diferentes bases numéricas entre las que el usuario puede elegir: Binario (2), Octal (8), Decimal (10), y Hexadecimal (16).
- `baseInput = new JComboBox<>(bases);` es la caja de selección desplegable que permitirá al usuario elegir la base del número de entrada, asegurando que se utilice la base correcta en el proceso de conversión.
- `baseInputPanel.add(baseInput);` añade la caja de selección desplegable (`JComboBox`) al panel `baseInputPanel`.

Configuración del Panel de Base de Salida

Ilustración 10 Panel de Base de Salida

```
47 // Configurar el color de fondo y márgenes del panel de base de salida
48 JPanel baseOutputPanel = new JPanel();
49 baseOutputPanel.setBackground(new Color( r:135, g:206, b:250)); // Color azul
50 baseOutputPanel.setBorder( border:BorderFactory.createCompoundBorder(
51     new LineBorder( color:Color.BLUE, thickness:5), // Borde azul de 5 píxeles
52     new EmptyBorder( top:10, left:10, bottom:10, right:10) // Margen interno
53 ));
54 baseOutputPanel.add(new JLabel( text:"Base de salida:"));
55 baseOutput = new JComboBox<>( items:bases);
56 baseOutputPanel.add( comp:baseOutput);
```

- `JPanel baseOutputPanel = new JPanel();` es el panel específico donde se colocan la etiqueta y la caja de selección desplegable para que el usuario elija la base de salida del número convertido.
- `baseOutputPanel.setBackground(new Color(135, 206, 250));` // Color azul, facilita la identificación de este panel como el lugar donde se selecciona la base de salida.
- `baseOutputPanel.setBorder(BorderFactory.createCompoundBorder(`
 `new LineBorder(Color.BLUE, 5),` // Borde azul de 5 píxeles
 `new EmptyBorder(10, 10, 10, 10)` // Margen interno
); el borde azul de 5 píxeles resalta el panel, mientras que el margen interno proporciona espacio adicional para evitar que los componentes estén demasiado cerca del borde, mejorando la estética y legibilidad del panel.
- `baseOutputPanel.add(new JLabel("Base de salida:"));` informa al usuario que debe seleccionar la base numérica en la que desea convertir el número ingresado, ayudando a guiar la interacción con la interfaz.

Cambio de Base

- `baseOutput = new JComboBox<>(bases);` es la caja de selección desplegable que proporciona al usuario las opciones de bases numéricas para la conversión del número de entrada.
- `baseOutputPanel.add(baseOutput);` aparece un cuadro de selección de base de salida en el panel junto a la etiqueta, lo que proporciona una interfaz clara y conveniente para seleccionar la base de salida.

Configuración del Panel de Botones

Ilustración 11 Panel de Botones

```
58 // Panel para los botones
59 JPanel buttonPanel = new JPanel();
60 buttonPanel.setLayout(new FlowLayout( align:FlowLayout.CENTER, hgap:10, vgap:10));
61
```

Se crea un panel para los botones con un `FlowLayout` que alinea los botones en el centro con un espacio de 10 píxeles entre ellos.

Ilustración 12 Botón de Conversión

```
62 // Botón para realizar la conversión
63 convertButton = new JButton( text:"Convertir");
64 convertButton.setPreferredSize(new Dimension( width:100, height:30)); // Tamaño del botón
65 convertButton.addActionListener( l:this);
66 buttonPanel.add( comp:convertButton);
```

Creamos el botón “Convertir”, donde `convertButton.setPreferredSize(new Dimension(100, 30));` define el tamaño preferido(del desarrollador), agregamos `convertButton.addActionListener(this);` para tener un mejor control sobre los clics, y añadimos el botón al panel con ayuda de `buttonPanel.add(convertButton);` .

Ilustración 13 Botón de Limpiar

```
68 // Botón para limpiar
69 clearButton = new JButton( text:"Limpiar");
70 clearButton.setPreferredSize(new Dimension( width:100, height:30)); // Tamaño del botón
71 clearButton.addActionListener( l:this);
72 buttonPanel.add( comp:clearButton);
```

De igual manera que definimos el botón de “Convertir”, utilizamos el mismo modo para crear el botón de “Limpiar”.

Configuración del Área de Salida

Ilustración 14 Área de salida

```
74 |  
75 |  
76 |  
-- |  
// Área de salida  
outputArea = new JTextArea();  
outputArea.setEditable(false);
```

- `outputArea = new JTextArea();` es en donde se mostrará el texto con el resultado de la conversión numérica. Al ser un área de texto, puede contener múltiples líneas, lo que permite que el resultado sea visible en un formato limpio y legible.
- `outputArea.setEditable(false);` esta configuración garantiza que `outputArea` se mantenga como un área de salida de solo lectura, donde el usuario no puede modificar o alterar el texto mostrado, lo que ayuda a prevenir errores o confusiones.

Componentes a las ventanas

Ilustración 15 Nuevos paneles al marco

```
78 |  
79 |  
80 |  
81 |  
82 |  
83 |  
84 |  
85 |  
86 |  
// Añadir los paneles al marco  
add(comp:inputPanel);  
add(comp:baseInputPanel);  
add(comp:baseOutputPanel);  
add(comp:buttonPanel); // Añadido el panel de botones  
add(new JScrollPane(view:outputArea));  
  
setVisible(b:true);  
}
```

- `add(inputPanel);` es el panel donde el usuario ingresa el número que desea convertir.
- `add(baseInputPanel);` es el panel donde el usuario selecciona la base numérica de entrada (por ejemplo, Binario, Octal, Decimal, Hexadecimal).
- `add(baseOutputPanel);` es el panel donde el usuario selecciona la base numérica de salida (en qué base quiere ver el resultado de la conversión).
- `add(buttonPanel); // Añadido el panel de botones,` es el panel que contiene los botones de "Convertir" y "Limpiar".
- `add(new JScrollPane(outputArea));` es el área donde se muestra el resultado de la conversión.

Cambio de Base

- setVisible(true); Hace que la ventana sea visible.

Manejo de Eventos

Ilustración 16 Eventos

```
88      @Override
89      public void actionPerformed(ActionEvent e) {
90          if (e.getSource() == clearButton) {
91              // Limpiar los campos de entrada y salida
92              inputField.setText("");
93              outputArea.setText("");
94          } else {
95              try {
96                  String input = inputField.getText();
97                  int inputBase = getBase((String) baseInput.getSelectedItem());
98                  int outputBase = getBase((String) baseOutput.getSelectedItem());
99
100                  int decimalValue = Integer.parseInt(input, inputBase);
101                  String result = Integer.toString(decimalValue, outputBase).toUpperCase();
102
103                  outputArea.setText("Resultado: " + result);
104              } catch (NumberFormatException ex) {
105                  outputArea.setText("Error: Entrada no válida.");
106              }
107          }
108      }
```

- @Override
public void actionPerformed(ActionEvent e) { : método actionPerformed es aquel que se ejecuta en respuesta a eventos de acción, como los clics en botones.
- if (e.getSource() == clearButton) {
 inputField.setText("");
 outputArea.setText(""); : verifican si el evento proviene del botón “Limpiar”. Si es así, limpia el contenido del campo de entrada y el área de salida.
- } else {
 try {
 String input = inputField.getText();
 int inputBase = getBase((String) baseInput.getSelectedItem());
 int outputBase = getBase((String) baseOutput.getSelectedItem()); : sí es evento proviene del botón “Convertir”, obtiene el texto del campo de entrada y las bases seleccionadas. Llama al método “getBase” para obtener los valores numéricos correspondientes a las bases seleccionadas.
- int decimalValue = Integer.parseInt(input, inputBase);
 String result = Integer.toString(decimalValue, outputBase).toUpperCase(); : se encargan de convertir el número de la base de entrada a decimal y luego de

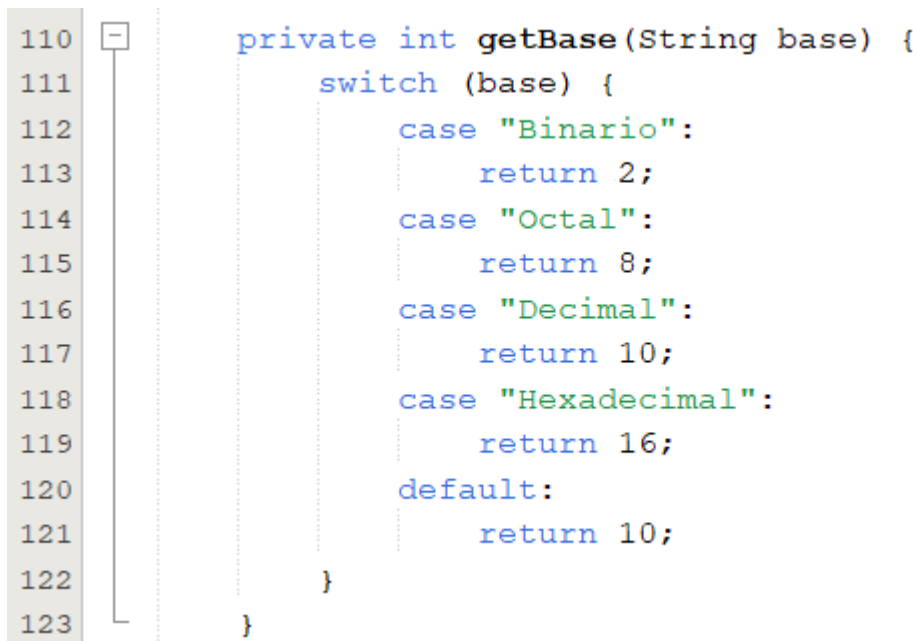
decimal a la base de salida seleccionada. Convierte el resultado a mayúsculas para mostrarlo.

- `outputArea.setText("Resultado: " + result);` : Muestra el resultado en el área de salida.
- `} catch (NumberFormatException ex) {`
 `outputArea.setText("Error: Entrada no válida.");`
 `}` : maneja excepciones en caso de errores de formato en la entrada del número. Muestra un mensaje de error en el área de salida si ocurre una excepción.

Método GetBase

Este método es esencial para la funcionalidad del programa. Cuando el usuario selecciona una base numérica en la interfaz, `getBase` convierte la selección en un número entero que el programa utiliza para realizar cálculos de conversión de bases. Sin este método, el programa no podría determinar cómo interpretar el número ingresado o a qué base convertirlo.

Ilustración 17 Método getBase



```
110 private int getBase(String base) {
111     switch (base) {
112         case "Binario":
113             return 2;
114         case "Octal":
115             return 8;
116         case "Decimal":
117             return 10;
118         case "Hexadecimal":
119             return 16;
120         default:
121             return 10;
122     }
123 }
```

- `private int getBase(String base) {` : convierte el nombre de la base numérica seleccionada a su valor numérico correspondiente, lo cual es esencial para realizar conversiones entre diferentes bases.

- switch (base) { : la estructura del interruptor simplifica el proceso de validación de una biblioteca/base de números seleccionada por el usuario, permitiendo que se ejecute el código apropiado para cada situación.
- case "Binario":
return 2; : Si el valor base es 'Binario', el método devuelve el valor 2. Cuando el usuario selecciona "Binario" en JComboBox, el programa interpreta el número ingresado como un número de base 2.
- case "Octal":
return 8; : cuando el usuario selecciona "Octal" en el JComboBox, el programa interpretará el número ingresado como un número en base 8.
- case "Decimal":
return 10; : cuando el usuario selecciona "Decimal" en el JComboBox, el programa interpreta el número ingresado como un número en base 10.
- case "Hexadecimal":
return 16; : este caso maneja la conversión de la cadena "Hexadecimal" al valor numérico 16, que es la base del sistema hexadecimal.
- default:
return 10; : aunque el usuario debería seleccionar una de las opciones válidas ("Binario", "Octal", "Decimal", "Hexadecimal"), este valor por defecto asegura que el programa no falle inesperadamente.

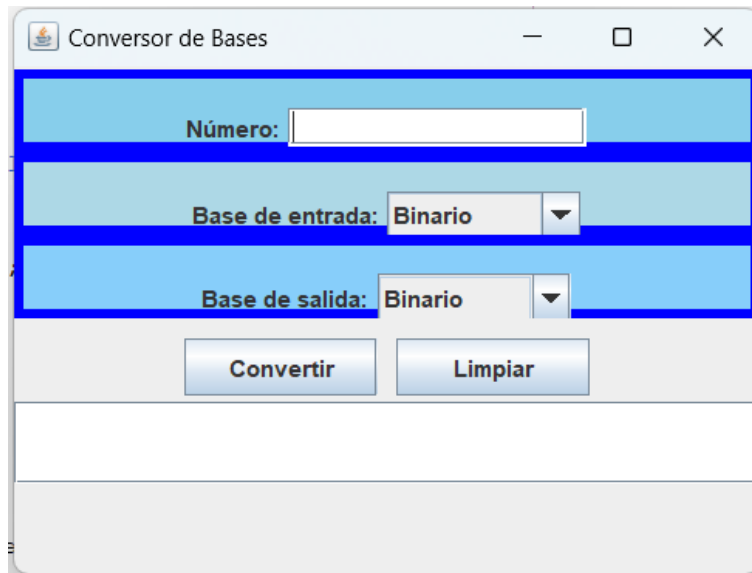
Método Main

Ilustración 18 Método main

```
125  [ ]  
126  |  
127  |  
128  |  
129  |  
125  public static void main(String[] args) {  
126      SwingUtilities.invokeLater(Cambio_de_Base::new);  
127  }  
128  }  
129  }
```

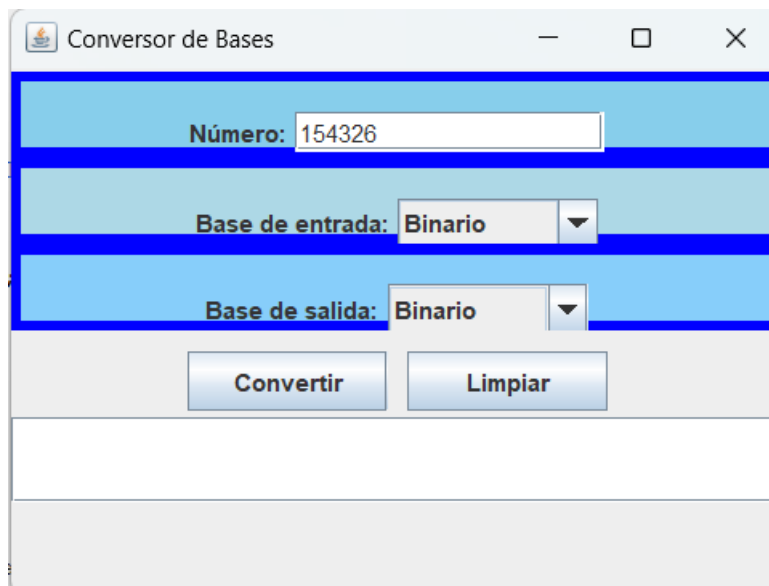
Defina el método principal, que es el punto de entrada de la aplicación. Utilice `SwingUtilities.invokeLater` para garantizar que la creación y visualización de la GUI se produzca en el hilo del evento Swing.

Ilustración 19 Panel Principal



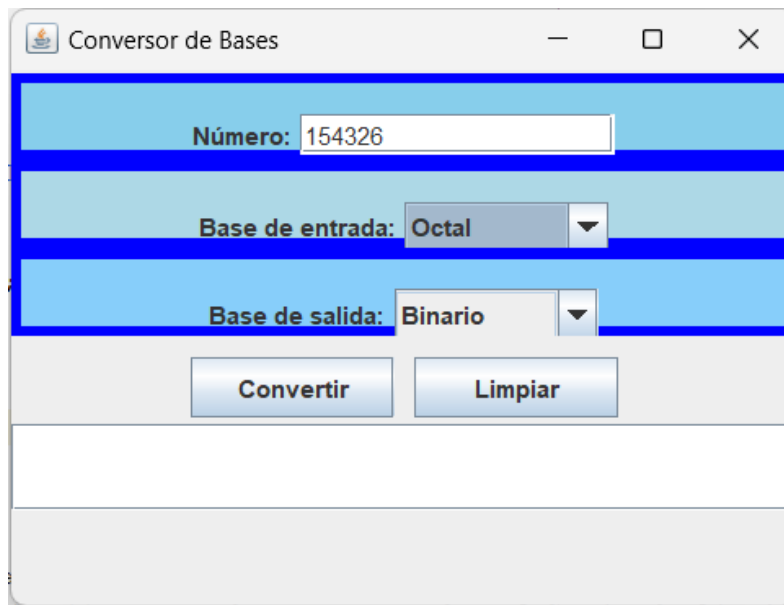
The screenshot shows a window titled "Conversor de Bases". It features three input fields with blue borders: "Número:" (empty), "Base de entrada:" (set to "Binario"), and "Base de salida:" (set to "Binario"). Below these are two buttons: "Convertir" and "Limpiar". A large empty white box is at the bottom for the result.

Ilustración 20 Número que deseamos convertir



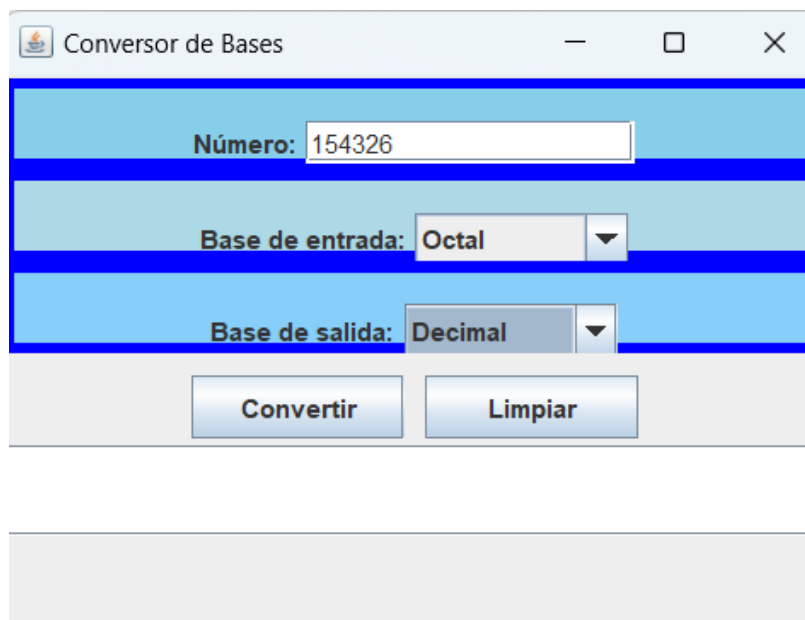
This screenshot is identical to the previous one, but the "Número:" input field now contains the text "154326".

Ilustración 21 Número a base de entrada



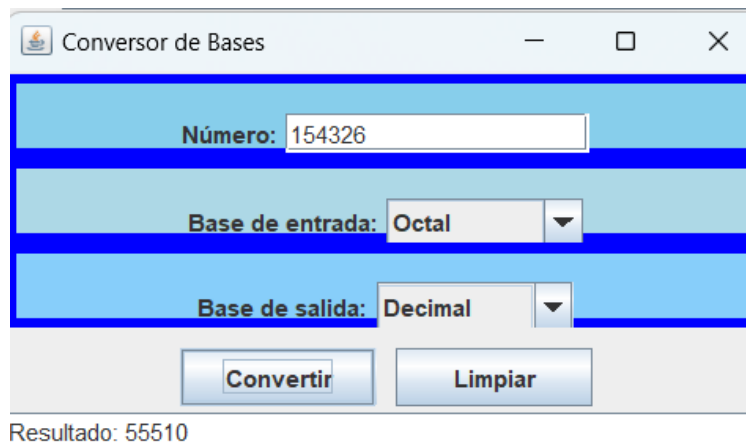
The screenshot shows a window titled "Conversor de Bases". It has a text input field labeled "Número:" containing the value "154326". Below this, there are two dropdown menus: "Base de entrada:" set to "Octal" and "Base de salida:" set to "Binario". At the bottom, there are two buttons: "Convertir" and "Limpiar". The input field and the "Base de entrada:" dropdown are highlighted with a blue border.

Ilustración 22 Número a base de salida



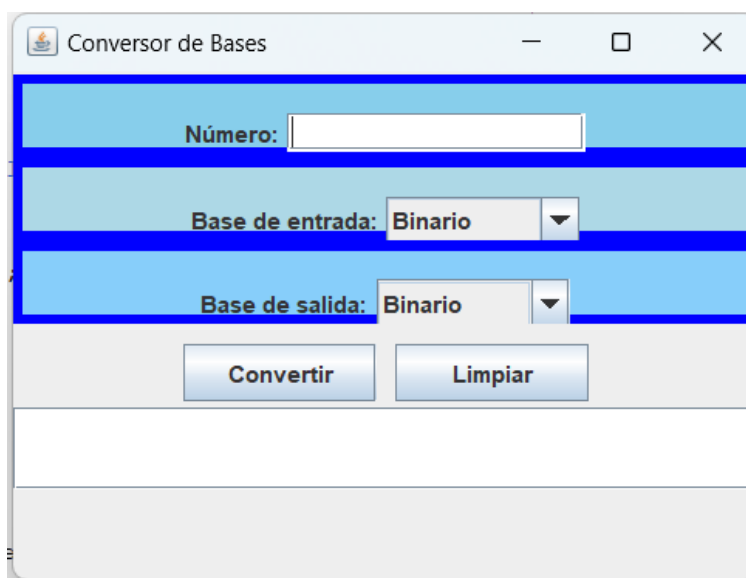
The screenshot shows the same "Conversor de Bases" window. The "Número:" field still contains "154326". The "Base de entrada:" dropdown is still set to "Octal", but the "Base de salida:" dropdown is now set to "Decimal". The "Convertir" and "Limpiar" buttons are still present. The input field and the "Base de salida:" dropdown are highlighted with a blue border.

Ilustración 23 Resultado de conversión



The screenshot shows a window titled "Conversor de Bases". It has three input fields: "Número:" with the value "154326", "Base de entrada:" with a dropdown menu set to "Octal", and "Base de salida:" with a dropdown menu set to "Decimal". Below these fields are two buttons: "Convertir" and "Limpiar". Below the buttons, the text "Resultado: 55510" is displayed.

Ilustración 24 Botón Limpiar



The screenshot shows the same "Conversor de Bases" window. The "Número:" field is now empty. The "Base de entrada:" dropdown is set to "Binario" and the "Base de salida:" dropdown is also set to "Binario". The "Convertir" and "Limpiar" buttons are still present. The "Resultado:" field is also empty.

Conclusión

Este programa fue creado para permitirnos explorar y aplicar conceptos clave en el diseño y manipulación de sistemas digitales. A lo largo del proceso de desarrollo, se ha demostrado la importancia de una planificación cuidadosa y el uso de herramientas como diagramas de flujo para guiar la implementación de soluciones efectivas. Este proceso no sólo produce programas funcionales que automatizan la conversión de bases numéricas, sino que también mejora la comprensión de cómo diseñar y organizar programas para obtener resultados óptimos. Esta experiencia enfatiza la importancia de la preparación y el análisis en el ciclo de desarrollo de software, asegurando que cada fase se ejecute de manera precisa y clara.