Report On

Currency Converter

Submitted in partial fulfillment of the requirements of the Course project in Semester III of Second Year Artificial Intelligence and Data Science

by

Umang Borse(03)

Ankush Chavate(05)

Atharva Chiplunkar(06)

Supervisor

Mrs. Sejal D'Mello

University of Mumbai

Vidyavardhini's College of Engineering & Technology

Department of Artificial Intelligence and Data Science

(2023-24)

Vidyavardhini's College of Engineering & Technology
Department of Artificial Intelligence and Data Science

CERTIFICATE

This is to certify that the project entitled “TIC-TAC-TOE-THE GAME” is a bonafide work of " Gautam Chaudhari(04), Mohammed Ali Jaffari(16), Ayush Gupta(14) " submitted to the University of Mumbai in partial fulfillment of the requirement for the Course project in semester III of Second Year Artificial Intelligence and Data Science engineering.

Supervisor

Mrs. Sejal D’Mello

Dr. Tatwadarshi P. N.
Head of Department

Table of Contents

Pg. No

1.OVERVIEW

Title: Currency Converter

1.1 Overview:

In our increasingly interconnected and globalized world, the necessity of converting one currency into another has become a routine aspect of modern life. This need arises for a multitude of reasons, spanning from personal travel and international trade to investment in foreign markets. In response to this demand, currency converters have emerged as invaluable tools that facilitate

and streamline the often complex and dynamic process of currency conversion.

A currency converter, at its core, serves as a practical and user-friendly solution for individuals and businesses seeking to navigate the ever-changing landscape of global currencies. It operates by providing access to precise and up-to-the-minute exchange rate information, which is fundamental for calculating the value of one currency in terms of another. By offering real-time exchange rates, these tools empower users to make informed financial decisions and to accurately gauge the economic impact of their transactions.

1.2 HOW TO PLAY: -

To use the currency converter app provided in the code you shared, follow these steps:

1. Launch the Application:

- Run the Java program, and the currency converter application window should appear.

2. Input the Amount:

In the "Amount:" text field, enter the amount of money you want to convert. Ensure that you enter a numerical value (e.g., 100).

3. Select Source Currency:

- Use the "From:" drop-down (JComboBox) to select the source currency. The available currency options include USD, EUR, JPY, GBP, CAD, AUD, CHF, CNY, and INR.

4. Select Target Currency:

- Use the "To:" drop-down (JComboBox) to select the target currency. You can choose from the same list of currencies.

5. Perform Conversion:

- After entering the amount and selecting both source and target currencies, click the "Convert" button. The application will calculate and display the converted amount in the "Result" label below the button.

6. Review Conversion Result:

- The result will be displayed in the format "X.XX Currency," where "X.XX" is the converted amount, and "Currency" is the target currency you selected.

7. Invalid Input Handling:

- If you enter non-numeric values or make invalid selections, such as choosing the same currency for both source and target, the "Result" label will display "Invalid input."

8. Repeat for Multiple Conversions:

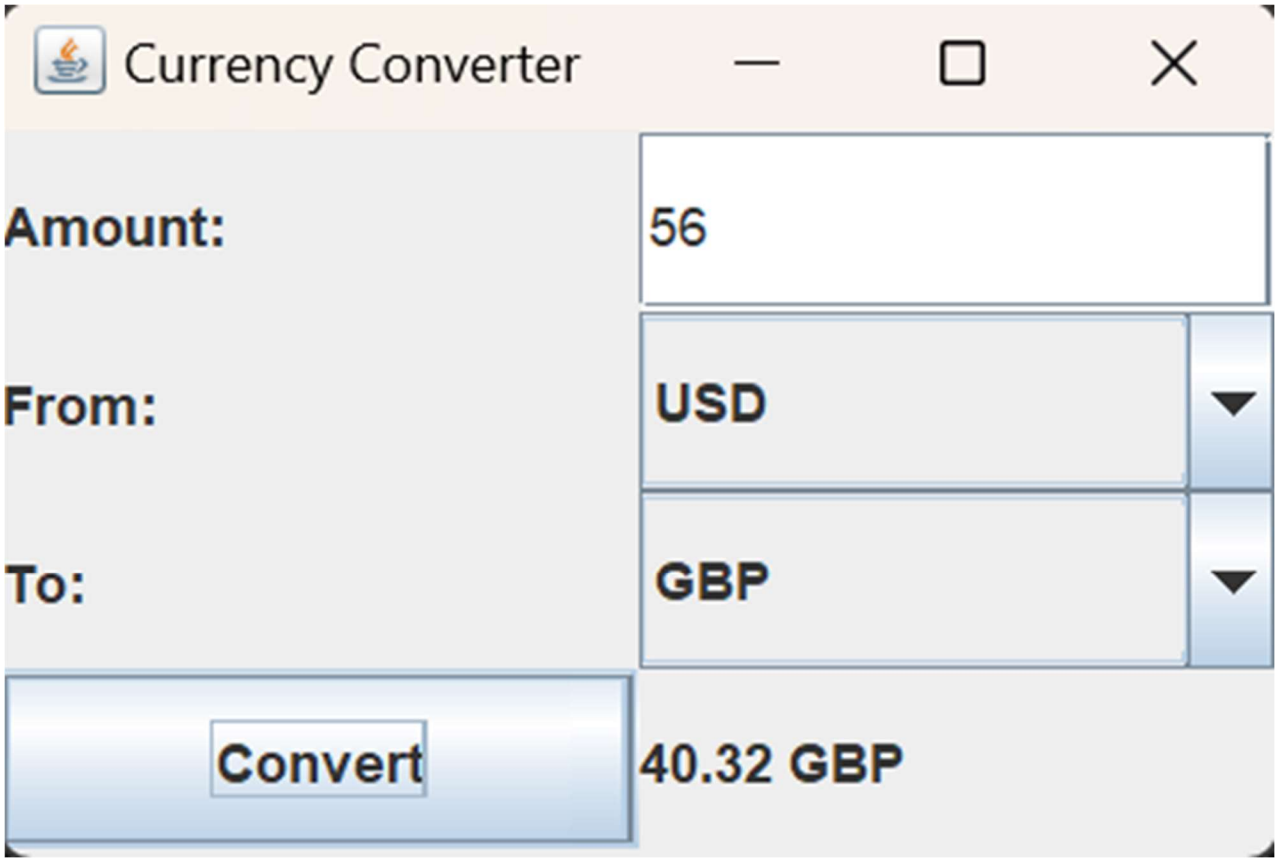
- You can perform multiple currency conversions by entering a new amount, selecting different source and target currencies, and clicking the "Convert" button again.

9. Close the Application:

- When you're finished using the currency converter, you can close the application window.

This currency converter app provides a straightforward way to convert between different currencies using predefined exchange rates. It's a simple tool that can be useful for quick currency conversions and educational purposes.

1.3 SNEAK – PEEK:



2.PROGRAM:-

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import
java.awt.event.ActionListener;
import java.text.DecimalFormat;

public class Currency_Converter
extends JFrame {

   private JLabel amountLabel,
fromLabel, toLabel, resultLabel;
   private JTextField amountField;
   private JComboBox<String>
fromComboBox, toComboBox;
   private JButton convertButton;
   private DecimalFormat
decimalFormat = new
DecimalFormat("#,##0.00");

   private final String[] currencies =
{"USD", "EUR", "JPY", "GBP",
"CAD", "AUD", "CHF", "CNY",
"INR"};
   private double[] exchangeRates =
{1.00, 0.84, 109.65, 0.72, 1.27, 1.30,
0.92, 6.47, 87.14};

   public Currency_Converter() {
      setTitle("Currency Converter");
      setLayout(new GridLayout(4,
2));

      amountLabel = new
JLabel("Amount:");
      add(amountLabel);

      amountField = new
JTextField();
      add(amountField);
```

```
        fromLabel = new JLabel("From:
");
        add(fromLabel);
        fromComboBox = new
JComboBox<>(currencies);
        add(fromComboBox);

        toLabel = new JLabel("To:");
        add(toLabel);
        toComboBox = new
JComboBox<>(currencies);
        add(toComboBox);

        convertButton = new
JButton("Convert");
        add(convertButton);

        resultLabel = new JLabel();
        add(resultLabel);


convertButton.addActionListener(ne
w ActionListener() {
            @Override
            public void
actionPerformed(ActionEvent e) {
                try {
                    double amount =
Double.parseDouble(amountField.get
Text());
                    String fromCurrency =
(String)
fromComboBox.getSelectedItem();
                    String toCurrency =
(String)
toComboBox.getSelectedItem();

                    double exchangeRate =
exchangeRates[getIndex(toCurrency)
] /
exchangeRates[getIndex(fromCurren
cy)];
                    double result = amount *
exchangeRate;
```

```
resultLabel.setText(decimalFormat.fo
rmat(result) + " " + toCurrency);
                } catch (Exception ex) {

resultLabel.setText("Invalid input");
                }
            }
        });

        setSize(300, 200);
        setVisible(true);

setDefaultCloseOperation(JFrame.E
XIT_ON_CLOSE);
    }

    private int getIndex(String
currency) {
        for (int i = 0; i <
currencies.length; i++) {
            if
(currency.equals(currencies[i])) {
                return i;
            }
        }
        return -1;
    }

    public static void main(String[]
args) {
        new Currency_Converter();
    }
}
```

3.TECHNICALITIES

3.1 TECHNOLOGIES USED:

Currency_Converter Class:

This class extends JFrame, making it a GUI application.

Instance Variables:

Several instance variables are declared to hold GUI components and data, including labels, text fields, combo boxes, a button, and exchange rate information.

Constructor (Currency_Converter):

In the constructor, the GUI layout is set to a 4x2 grid, which organizes the components on the interface.

GUI Components:

amountLabel, fromLabel, and toLabel are JLabel components that display text.

amountField is a JTextField for entering the amount to convert.

fromComboBox and toComboBox are JComboBox components containing currency options.

convertButton is a JButton used to initiate the conversion process.

resultLabel is another JLabel to display the conversion result.

DecimalFormat:

decimalFormat is used to format the conversion result with two decimal places.

Arrays for Currencies and Exchange Rates:

currencies and exchangeRates are arrays that store currency codes and their corresponding exchange rates. These values can be expanded or modified as needed.

ActionListener for convertButton:

An ActionListener is added to the convertButton to handle the conversion process. When the button is clicked, the code inside the actionPerformed method is executed.

It parses the amount entered in the amountField as a double and determines the source (fromCurrency) and target (toCurrency) currencies selected in the combo boxes.

The conversion is performed using the exchange rates and displayed in the resultLabel. If an exception occurs during parsing or if the currencies are not found, it displays "Invalid input."

getIndex Method:

getIndex is a helper method to retrieve the index of a currency code in the currencies array. It is used to find the index of the selected currencies for exchange rate calculation.

Main Method:

The main method creates an instance of the Currency_Converter class, which initializes the GUI application.

GUI Configuration:

The size of the GUI window is set to 300x200 pixels.

The window is made visible (setVisible(true)) and configured to exit the application when closed (setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE)).

3.2 EXPLANATION:-

1. The code begins by importing the necessary Java libraries, including Swing for GUI components and event handling, AWT for basic GUI functionality, and Random for generating random numbers.
2. Instance Variables:

   Several instance variables are declared to hold GUI components and data, including labels, text fields, combo boxes, a button, and exchange rate information.
3. Constructor (Currency_Converter):

   In the constructor, the GUI layout is set to a 4x2 grid, which organizes the components on the interface.
4. GUI Components:

   amountLabel, fromLabel, and toLabel are JLabel components that display text.

   amountField is a JTextField for entering the amount to convert.

   From ComboBox and toComboBox are JComboBox components containing currency options.
5. convertButton is a JButton used to initiate the conversion process.

   resultLabel is another JLabel to display the conversion result.
6. DecimalFormat:

   decimalFormat is used to format the conversion result with two decimal places.
7. Arrays for Currencies and Exchange Rates:

   Currencies and exchangeRates are arrays that store currency codes and their corresponding exchange rates. These values can be expanded or modified as needed.

8. ActionListener for convertButton:

   An ActionListener is added to the convertButton to handle the conversion process. When the button is clicked, the code inside the actionPerformed method is executed.

   It parses the amount entered in the amountField as a double and determines the source (fromCurrency) and target (toCurrency) currencies selected in the combo boxes.

   The conversion is performed using the exchange rates and displayed in the resultLabel. If an exception occurs during parsing or if the currencies are not found, it displays "Invalid input."

9. getIndex Method:

   getIndex is a helper method to retrieve the index of a currency code in the currencies array. It is used to find the index of the selected currencies for exchange rate calculation.

10. Main Method:

    The main method creates an instance of the Currency_Converter class, which initializes the GUI application.

    GUI Configuration:

    The size of the GUI window is set to 300x200 pixels.

    The window is made visible (setVisible(true)) and configured to exit the application when closed (setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE)).

4. CONCLUSION:

In conclusion, the Currency Converter app stands as a valuable tool that simplifies the complex process of currency conversion in our globalized economy. Its user-friendly interface, real-time data retrieval, and educational value make it a versatile and accessible solution for a wide range of users. Whether for travel, commerce, or education, this application equips users with the means to navigate the intricate world of foreign exchange with confidence and ease.