

**JAVA PROJECT**

***CURRENCY CONVERTOR***

**Submitted by –**

Name : Anamika Barpagga Roll no : RK21YKB66

Registration Number : 12109422

Section : Yk

Name : Anurag Singh

Roll no : RK21YK B60

Registration No : 12111401

Section : Yk

Name : Kotha Rohith Gupta

Roll no : RK21YKA35

Registration No : 12105729

Section : Yk

INTRODUCTION

The **Currency Converter Project In Java** is a piece of software that allows you to convert currency values quickly and easily using current exchange rates. These currency converters are commonly accessed for free on the Internet, and they can swiftly convert the value of one currency to another, such as dollars to peso or peso to pounds, and back.

INTERFACE

APIs are important software components bundled with the JDK. APIs in Java include classes, interfaces, and user Interfaces. They enable developers to integrate various applications and websites and offer real-time information.

* **AWT API**

**Java AWT** (Abstract Window Toolkit) is an API to develop Graphical User Interface (GUI) or windows-based applications in Java.

Java AWT components are platform-dependent i.e. components are displayed according to the view of operating system. AWT is heavy weight i.e. its components are using the resources of underlying operating system (OS).

The java.awt [package](https://www.javatpoint.com/package) provides [classes](https://www.javatpoint.com/object-and-class-in-java) for AWT API such as [TextField](https://www.javatpoint.com/java-awt-textfield), [Label](https://www.javatpoint.com/java-awt-label), [TextArea](https://www.javatpoint.com/java-awt-textarea), RadioButton, [CheckBox](https://www.javatpoint.com/java-awt-checkbox), [Choice](https://www.javatpoint.com/java-awt-choice), [List](https://www.javatpoint.com/java-awt-list) etc.

* **SWING APT**

**Java Swing tutorial** is a part of Java Foundation Classes (JFC) that is used to create window-based applications. It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java.

Unlike AWT, Java Swing provides platform-independent and lightweight components.

The javax.swing package provides classes for java swing API such as JButton, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser etc.

CODE:

* **About Window:**

package currencyConverter;

import java.awt.Color;

import java.awt.Cursor;

import java.awt.Desktop;

import java.awt.Font;

import java.awt.event.MouseAdapter;

import java.awt.event.MouseEvent;

import javax.swing.border.EmptyBorder;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

import javax.swing.SwingConstants;

import java.net.URI;

import java.util.ResourceBundle;

public class AboutWindow extends JFrame {

private static final ResourceBundle BUNDLE = ResourceBundle.getBundle("localization.translation"); //$NON-NLS-1$

private JPanel contentPane;

private static AboutWindow windowInstance = null;

/\*\*

\* Create the aboutWindow frame

\*/

private AboutWindow() {

setTitle(BUNDLE.getString("AboutWindow.this.title")); //$NON-NLS-1$

setBounds(100, 100, 347, 260);

setLocationRelativeTo(null);

setResizable( false );

// Window components

contentPane = new JPanel();

contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));

setContentPane(contentPane);

// Label "author"

JLabel lblAuthor = new JLabel(BUNDLE.getString("AboutWindow.lblAuthor.text")); //$NON-NLS-1$

lblAuthor.setHorizontalAlignment(SwingConstants.CENTER);

lblAuthor.setBounds(65, 122, 219, 33);

// label with a clickable link

JLabel lblLink = new JLabel("http://jvinceno.fr");

lblLink.setBounds(65, 159, 219, 33);

lblLink.setHorizontalAlignment(SwingConstants.CENTER);

lblLink.setForeground(Color.BLUE);

lblLink.setCursor(new Cursor(Cursor.HAND\_CURSOR));

lblLink.addMouseListener(new MouseAdapter() {

@Override

public void mouseClicked(MouseEvent e) {

try {

Desktop.getDesktop().browse(new URI("http://www.jvinceno.fr"));

} catch (Exception ex) {

ex.printStackTrace();

}

}

});

contentPane.setLayout(null);

// label "title"

JLabel lblTitle = new JLabel("Currency Converter");

lblTitle.setBounds(65, 12, 219, 33);

lblTitle.setHorizontalAlignment(SwingConstants.CENTER);

lblTitle.setFont(new Font("Noto Sans", Font.BOLD, 15));

contentPane.add(lblTitle);

// label "licence"

JLabel lblLicenceMit = new JLabel("Licence GNU GPL v3.0");

lblLicenceMit.setBounds(65, 77, 219, 33);

lblLicenceMit.setHorizontalAlignment(SwingConstants.CENTER);

contentPane.add(lblLicenceMit);

// label "version"

JLabel lblVersion = new JLabel("Version 1.0");

lblVersion.setHorizontalAlignment(SwingConstants.CENTER);

lblVersion.setBounds(65, 45, 219, 33);

contentPane.add(lblVersion);

contentPane.add(lblAuthor);

contentPane.add(lblLink);

}

// Check if the window is already created to launch only one instance of the window.

public static AboutWindow getInstance() {

if (windowInstance == null) {

windowInstance = new AboutWindow();

}

return windowInstance;

}

}

* **Main Window:**

package currencyConverter;

import java.awt.EventQueue;

import javax.swing.JFrame;

import javax.swing.JPanel;

import javax.swing.border.EmptyBorder;

import javax.swing.JLabel;

import javax.swing.JComboBox;

import javax.swing.JTextField;

import javax.swing.JButton;

import javax.swing.JMenuBar;

import javax.swing.JMenu;

import javax.swing.JMenuItem;

import java.awt.event.ActionListener;

import java.awt.event.KeyEvent;

import java.text.DecimalFormat;

import java.util.ArrayList;

import java.awt.event.ActionEvent;

import javax.swing.SwingConstants;

import java.util.ResourceBundle;

public class MainWindow extends JFrame {

private static final ResourceBundle BUNDLE = ResourceBundle.getBundle("localization.translation"); //$NON-NLS-1$

private JPanel contentPane;

private JTextField fieldAmount;

private ArrayList<Currency> currencies = Currency.init();

/\*\*

\* Create the mainWindow frame

\*/

public MainWindow() {

setTitle(BUNDLE.getString("MainWindow.this.title")); //$NON-NLS-1$

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setBounds(100, 100, 589, 300);

setLocationRelativeTo(null);

setResizable( false );

// Create menu bar

JMenuBar menuBar = new JMenuBar();

setJMenuBar(menuBar);

// "File" menu

JMenu mnFile = new JMenu(BUNDLE.getString("MainWindow.mnFile.text")); //$NON-NLS-1$

mnFile.setMnemonic(KeyEvent.VK\_F);

menuBar.add(mnFile);

// "Quit" menu item

JMenuItem mntmQuit = new JMenuItem(BUNDLE.getString("MainWindow.mntmQuit.text")); //$NON-NLS-1$

mntmQuit.setMnemonic(KeyEvent.VK\_Q);

mntmQuit.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent arg0) {

System.exit(0);

}

});

mnFile.add(mntmQuit);

// "Help" menu

JMenu mnHelp = new JMenu(BUNDLE.getString("MainWindow.mnHelp.text")); //$NON-NLS-1$

mnHelp.setMnemonic(KeyEvent.VK\_H);

menuBar.add(mnHelp);

// "About" menu item

JMenuItem mntmAbout = new JMenuItem(BUNDLE.getString("MainWindow.mntmAbout.text")); //$NON-NLS-1$

mntmAbout.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent arg0) {

EventQueue.invokeLater(new Runnable() {

public void run() {

try {

AboutWindow.getInstance().setVisible(true);

} catch (Exception e) {

e.printStackTrace();

}

}

});

}

});

mntmAbout.setMnemonic(KeyEvent.VK\_A);

mnHelp.add(mntmAbout);

// Window components

contentPane = new JPanel();

contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));

setContentPane(contentPane);

contentPane.setLayout(null);

// Label "Convert"

JLabel lblConvert = new JLabel(BUNDLE.getString("MainWindow.lblConvert.text")); //$NON-NLS-1$

lblConvert.setHorizontalAlignment(SwingConstants.RIGHT);

lblConvert.setBounds(0, 14, 92, 15);

contentPane.add(lblConvert);

// ComboBox of the first currency

final JComboBox<String> comboBoxCountry1 = new JComboBox<String>();

comboBoxCountry1.setBounds(147, 7, 240, 28);

populate(comboBoxCountry1, currencies);

contentPane.add(comboBoxCountry1);

// Label "To"

JLabel lblTo = new JLabel(BUNDLE.getString("MainWindow.lblTo.text")); //$NON-NLS-1$

lblTo.setHorizontalAlignment(SwingConstants.RIGHT);

lblTo.setBounds(66, 54, 26, 15);

contentPane.add(lblTo);

// ComboBox of the second currency

final JComboBox<String> comboBoxCountry2 = new JComboBox<String>();

comboBoxCountry2.setBounds(147, 47, 240, 28);

populate(comboBoxCountry2, currencies);

contentPane.add(comboBoxCountry2);

// Label "Amount"

final JLabel lblAmount = new JLabel(BUNDLE.getString("MainWindow.lblAmount.text")); //$NON-NLS-1$

lblAmount.setHorizontalAlignment(SwingConstants.RIGHT);

lblAmount.setBounds(23, 108, 69, 15);

contentPane.add(lblAmount);

// Textfield where the user

fieldAmount = new JTextField();

fieldAmount.setText("0.00");

fieldAmount.setBounds(147, 101, 103, 29);

contentPane.add(fieldAmount);

fieldAmount.setColumns(10);

fieldAmount.setDocument(new JTextFieldLimit(8));

// Label displaying result of conversion

final JLabel lblResult = new JLabel("");

lblResult.setHorizontalAlignment(SwingConstants.LEFT);

lblResult.setBounds(147, 188, 428, 38);

contentPane.add(lblResult);

// Button "Convert"

JButton btnConvert = new JButton(BUNDLE.getString("MainWindow.btnConvert.text")); //$NON-NLS-1$

btnConvert.setBounds(147, 142, 129, 38);

btnConvert.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent arg0) {

String nameCurrency1 = comboBoxCountry1.getSelectedItem().toString();

String nameCurrency2 = comboBoxCountry2.getSelectedItem().toString();

String result;

String formattedPrice;

String formattedAmount;

Double price;

Double amount = 0.0;

DecimalFormat format = new DecimalFormat("#0.00");

try {

amount = Double.parseDouble( fieldAmount.getText() ) ;

} catch (NumberFormatException e) {

e.printStackTrace();

amount = 0.0;

}

price = convert(nameCurrency1, nameCurrency2, currencies, amount);

// Format numbers to avoid "E7" problem

formattedPrice = format.format(price);

formattedAmount = format.format(amount);

result = formattedAmount + " " + nameCurrency1 + " = " + formattedPrice + " " + nameCurrency2;

lblResult.setText(result);

}

});

contentPane.add(btnConvert);

}

// Fill comboBox with currencies name

public static void populate(JComboBox<String> comboBox, ArrayList<Currency> currencies) {

for (Integer i = 0; i < currencies.size(); i++) {

comboBox.addItem( currencies.get(i).getName() );

}

}

// Find the short name and the exchange value of the second currency

public static Double convert(String currency1, String currency2, ArrayList<Currency> currencies, Double amount) {

String shortNameCurrency2 = null;

Double exchangeValue;

Double price = 0.0;

// Find shortname for the second currency

for (Integer i = 0; i < currencies.size(); i++) {

if (currencies.get(i).getName() == currency2) {

shortNameCurrency2 = currencies.get(i).getShortName();

break;

}

}

// Find exchange value and call convert() to calcul the new price

if (shortNameCurrency2 != null) {

for (Integer i = 0; i < currencies.size(); i++) {

if (currencies.get(i).getName() == currency1) {

exchangeValue = currencies.get(i).getExchangeValues().get(shortNameCurrency2);

price = Currency.convert(amount, exchangeValue);

break;

}

}

}

return price;

}

}

* **Currency:**

package currencyConverter;

import java.util.ArrayList;

import java.util.HashMap;

public class Currency {

private String name;

private String shortName;

private HashMap<String, Double> exchangeValues = new HashMap<String, Double>();

// "Currency" Constructor

public Currency(String nameValue, String shortNameValue) {

this.name = nameValue;

this.shortName = shortNameValue;

}

// Getter for name

public String getName() {

return this.name;

}

// Setter for name

public void setName(String name) {

this.name = name;

}

// Getter for shortName

public String getShortName() {

return this.shortName;

}

// Setter for shortName

public void setShortName(String shortName) {

this.shortName = shortName;

}

// Getter for exchangeValues

public HashMap<String, Double> getExchangeValues() {

return this.exchangeValues;

}

// Setter for exchangeValues

public void setExchangeValues(String key, Double value) {

this.exchangeValues.put(key, value);

}

// Set default values for a currency

public void defaultValues() {

String currency = this.name;

switch (currency) {

case "US Dollar":

this.exchangeValues.put("USD", 1.00);

this.exchangeValues.put("EUR", 0.93);

this.exchangeValues.put("GBP", 0.66);

this.exchangeValues.put("CHF", 1.01);

this.exchangeValues.put("CNY", 6.36);

this.exchangeValues.put("JPY", 123.54);

break;

case "Euro":

this.exchangeValues.put("USD", 1.073);

this.exchangeValues.put("EUR", 1.00);

this.exchangeValues.put("GBP", 0.71);

this.exchangeValues.put("CHF", 1.08);

this.exchangeValues.put("CNY", 6.83);

this.exchangeValues.put("JPY", 132.57);

break;

case "British Pound":

this.exchangeValues.put("USD", 1.51);

this.exchangeValues.put("EUR", 1.41);

this.exchangeValues.put("GBP", 1.00);

this.exchangeValues.put("CHF", 1.52);

this.exchangeValues.put("CNY", 9.60);

this.exchangeValues.put("JPY", 186.41);

break;

case "Swiss Franc":

this.exchangeValues.put("USD", 0.99);

this.exchangeValues.put("EUR", 0.93);

this.exchangeValues.put("GBP", 0.66);

this.exchangeValues.put("CHF", 1.00);

this.exchangeValues.put("CNY", 6.33);

this.exchangeValues.put("JPY", 122.84);

break;

case "Chinese Yuan Renminbi":

this.exchangeValues.put("USD", 0.16);

this.exchangeValues.put("EUR", 0.15);

this.exchangeValues.put("GBP", 0.11);

this.exchangeValues.put("CHF", 0.16);

this.exchangeValues.put("CNY", 1.00);

this.exchangeValues.put("JPY", 19.41);

break;

case "Japanese Yen":

this.exchangeValues.put("USD", 0.008);

this.exchangeValues.put("EUR", 0.007);

this.exchangeValues.put("GBP", 0.005);

this.exchangeValues.put("CHF", 0.008);

this.exchangeValues.put("CNY", 0.051);

this.exchangeValues.put("JPY", 1.000);

break;

}

}

// Initialize currencies

public static ArrayList<Currency> init() {

ArrayList<Currency> currencies = new ArrayList<Currency>();

currencies.add( new Currency("US Dollar", "USD") );

currencies.add( new Currency("Euro", "EUR") );

currencies.add( new Currency("British Pound", "GBP") );

currencies.add( new Currency("Swiss Franc", "CHF") );

currencies.add( new Currency("Chinese Yuan Renminbi", "CNY") );

currencies.add( new Currency("Japanese Yen", "JPY") );

for (Integer i =0; i < currencies.size(); i++) {

currencies.get(i).defaultValues();

}

return currencies;

}

// Convert a currency to another

public static Double convert(Double amount, Double exchangeValue) {

Double price;

price = amount \* exchangeValue;

price = Math.round(price \* 100d) / 100d;

return price;

}

}

* **Currency Convertor:**

package currencyConverter;

import java.awt.EventQueue;

import javax.swing.UIManager;

public class CurrencyConverter {

public static void main(String[] args) {

try {

UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());

} catch (Throwable e) {

e.printStackTrace();

}

EventQueue.invokeLater(new Runnable() {

public void run() {

try {

// Create and show main window at startup

MainWindow mainWindow = new MainWindow();

mainWindow.setVisible(true);

} catch (Exception e) {

e.printStackTrace();

}

}

});

}

}

* **JTextFieldLimit:**

package currencyConverter;

import javax.swing.text.AttributeSet;

import javax.swing.text.BadLocationException;

import javax.swing.text.PlainDocument;

public class JTextFieldLimit extends PlainDocument {

private int limit;

private boolean toUppercase = false;

JTextFieldLimit(int limit) {

super();

this.limit = limit;

}

JTextFieldLimit(int limit, boolean upper) {

super();

this.limit = limit;

toUppercase = upper;

}

public void insertString

(int offset, String str, AttributeSet attr)

throws BadLocationException {

if (str == null) return;

if ((getLength() + str.length()) <= limit) {

if (toUppercase) str = str.toUpperCase();

super.insertString(offset, str, attr);

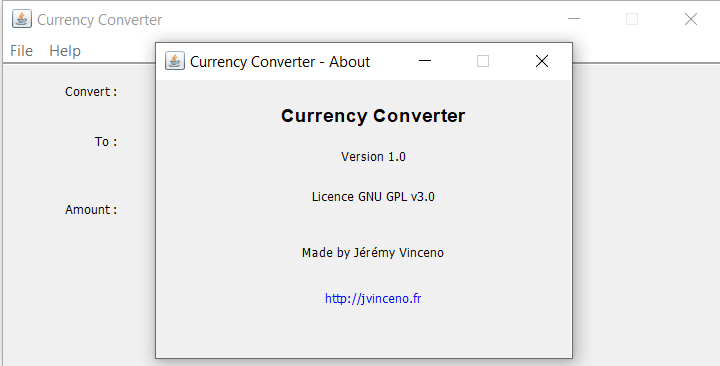
}

}

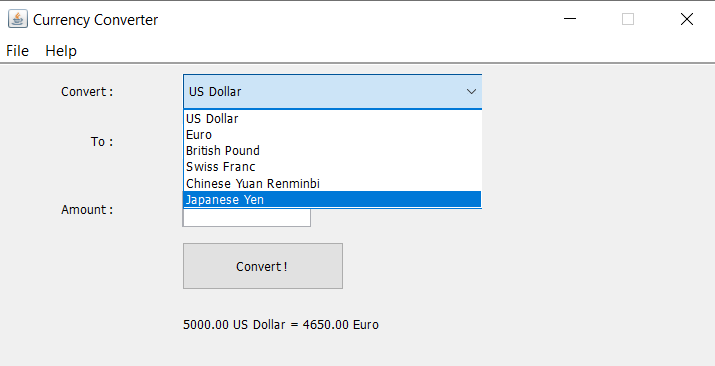
}

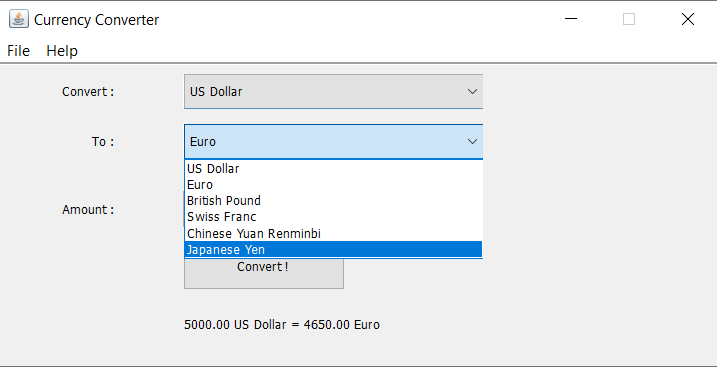
Execution:

**About window:**

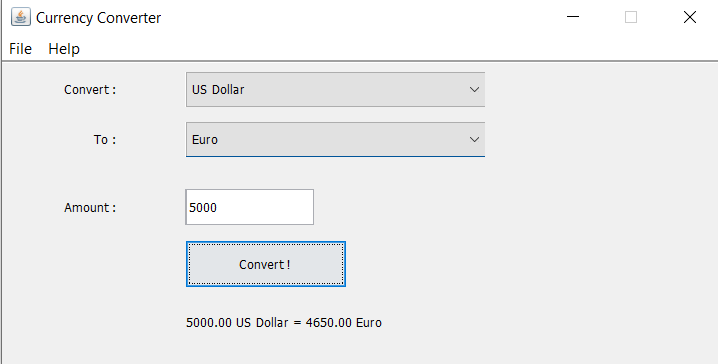


**Currency Convertor:**





**Final Execution:**



REFERENCES:

* Javatpoint