Group Members:

Muhammad Abdullah Aish (BSCS-FA18-03)

Reehum Farrukh (BSCS-FA18-06)

Umer Iqbal (BSCS-FA18-07)

Ahmed Sarfaraz (BSCS-FA18-16)

Currency Converter:

Currency converter (or currency exchange) is a project coded in Java programming language. This simple application provides a web-based interface for exchanging/converting money from one currency (say $) to another currency (say €).

Currency Converter Project Abstract:

Different countries use different currency, and there is daily variation in these currencies relative to one another. Those who transfer money from one country to another (one currency to another) must be updated with the latest currency exchange rates in the market.

Currency converter project is built keeping this thing in mind. It is simply a calculator-like app. In this application, there is regular update about currency of every country by which it displays present currency market value and conversion rate.

Such application can be used by any user, but it is mainly useful for business, shares, and finance related areas where money transfer and currency exchange takes place on a daily basis.

In this currency converter app, users are provided with an option to select the type of conversion, i.e. from “this” currency to “that” currency. This simple feature allows users to enter amount to be converted (say currency in Dollars), and display the converted amount (say currency in Euro).

Tool:

Eclipse IDE

Platform:

Java

Code:

AboutWindow.java:

```
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"); //

	private JPanel contentPane;

	private static AboutWindow windowInstance = null;



//		Create the aboutWindow frame


	private AboutWindow() {

		setTitle(BUNDLE.getString("AboutWindow.this.title"));

		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"));

		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;
	}
}
```

MainWindow.java:

```
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"); //

	private JPanel contentPane;

	private JTextField fieldAmount;

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


	/**

	 * Create the mainWindow frame

	 */

	public MainWindow() {

		setTitle(BUNDLE.getString("MainWindow.this.title")); //

		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")); //
                    mnFile.setMnemonic(KeyEvent.VK_F);
                    menuBar.add(mnFile);

                    // "Quit" menu item
                    JMenuItem mntmQuit = new
JMenuItem(BUNDLE.getString("MainWindow.mntmQuit.text")); //
                    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")); //
                    mnHelp.setMnemonic(KeyEvent.VK_H);
                    menuBar.add(mnHelp);

                    // "About" menu item
                    JMenuItem mntmAbout = new
JMenuItem(BUNDLE.getString("MainWindow.mntmAbout.text"));                    //
                    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")); //
        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")); //
        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"));
//
        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"));
//
        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);
```

```
                        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.java:

```
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", 0.84);
			this.exchangeValues.put("GBP", 0.75);
			this.exchangeValues.put("PK", 165.95);
			this.exchangeValues.put("CNY", 6.84);
			this.exchangeValues.put("JPY", 106.24);
			this.exchangeValues.put("CAD", 1.31);
			this.exchangeValues.put("AUD", 1.37);
			this.exchangeValues.put("CHF", 0.91);
			this.exchangeValues.put("NZD", 1.49);
			this.exchangeValues.put("KD", 0.31);
			break;
		case "Euro":
```

```
            this.exchangeValues.put("USD", 1.18);
            this.exchangeValues.put("EUR", 1.00);
            this.exchangeValues.put("GBP", 0.89);
            this.exchangeValues.put("PK", 196.46);
            this.exchangeValues.put("CNY", 8.10);
            this.exchangeValues.put("JPY", 125.77);
            this.exchangeValues.put("CAD", 0.88);
            this.exchangeValues.put("AUD", 0.92);
            this.exchangeValues.put("CHF", 0.61);
            this.exchangeValues.put("NZD", 1.000);
            this.exchangeValues.put("KD", 0.36);
            break;
    case "British Pound":
            this.exchangeValues.put("USD", 1.33);
            this.exchangeValues.put("EUR", 1.12);
            this.exchangeValues.put("GBP", 1.00);
            this.exchangeValues.put("PK", 220.54);
            this.exchangeValues.put("CNY", 9.09);
            this.exchangeValues.put("JPY", 141.19);
            this.exchangeValues.put("CAD", 1.55);
            this.exchangeValues.put("AUD", 1.63);
            this.exchangeValues.put("CHF", 1.08);
            this.exchangeValues.put("NZD", 1.77);
            this.exchangeValues.put("KD", 0.41);
            break;
    case "Pakistan":
            this.exchangeValues.put("USD", 0.0060);
            this.exchangeValues.put("EUR", 0.0051);
            this.exchangeValues.put("GBP", 0.0045);
            this.exchangeValues.put("PK", 1.00);
            this.exchangeValues.put("CNY", 0.041);
            this.exchangeValues.put("JPY", 0.64);
            this.exchangeValues.put("CAD", 0.0070);
            this.exchangeValues.put("AUD", 0.0083);
            this.exchangeValues.put("CHF", 0.0055);
            this.exchangeValues.put("NZD", 0.0090);
            this.exchangeValues.put("KD", 0.0018);
            break;
        case "Chinese Yuan Renminbi":
            this.exchangeValues.put("USD", 0.15);
            this.exchangeValues.put("EUR", 0.12);
            this.exchangeValues.put("GBP", 0.11);
            this.exchangeValues.put("PK", 24.25);
            this.exchangeValues.put("CNY", 1.00);
            this.exchangeValues.put("JPY", 15.53);
            this.exchangeValues.put("CAD", 0.19);
            this.exchangeValues.put("AUD", 0.20);
            this.exchangeValues.put("CHF", 0.13);
            this.exchangeValues.put("NZD", 0.22);
            this.exchangeValues.put("KD", 0.045);
            break;
        case "Japanese Yen":
            this.exchangeValues.put("USD", 0.0094);
            this.exchangeValues.put("EUR", 0.0080);
            this.exchangeValues.put("GBP", 0.0071);
            this.exchangeValues.put("PK", 1.56);
            this.exchangeValues.put("CNY", 0.064);
            this.exchangeValues.put("JPY", 1.000);
            this.exchangeValues.put("CAD", 0.012);
            this.exchangeValues.put("AUD", 0.013);
            this.exchangeValues.put("CHF", 0.0086);
            this.exchangeValues.put("KD", 0.0029);
            this.exchangeValues.put("NZD", 0.014);
            break;
        case "Canadian Dollar":
            this.exchangeValues.put("USD", 0.77);
            this.exchangeValues.put("EUR", 0.65);
            this.exchangeValues.put("GBP", 0.58);
            this.exchangeValues.put("PK", 127.04);
            this.exchangeValues.put("CNY", 5.24);
            this.exchangeValues.put("JPY", 81.33);
            this.exchangeValues.put("CAD", 1.000);
            this.exchangeValues.put("AUD", 1.05);
            this.exchangeValues.put("CHF", 0.70);
            this.exchangeValues.put("KD", 0.23);
            this.exchangeValues.put("NZD", 1.14);
            break;
        case "Australian Dollar":
            this.exchangeValues.put("USD", 0.73);
            this.exchangeValues.put("EUR", 0.62);
            this.exchangeValues.put("GBP", 0.55);
            this.exchangeValues.put("PK", 120.89);
            this.exchangeValues.put("CNY", 4.98);
            this.exchangeValues.put("JPY", 77.36);
            this.exchangeValues.put("CAD", 0.95);
            this.exchangeValues.put("AUD", 1.000);
            this.exchangeValues.put("CHF", 0.66);
```

```
            this.exchangeValues.put("KD", 0.22);
            this.exchangeValues.put("NZD", 1.09);
            break;
    case "Swiss Franc":
            this.exchangeValues.put("USD", 1.10);
            this.exchangeValues.put("EUR", 0.93);
            this.exchangeValues.put("GBP", 0.82);
            this.exchangeValues.put("PK", 181.79);
            this.exchangeValues.put("CNY", 7.50);
            this.exchangeValues.put("JPY", 116.38);
            this.exchangeValues.put("CAD", 1.43);
            this.exchangeValues.put("AUD", 1.50);
            this.exchangeValues.put("CHF", 1.000);
            this.exchangeValues.put("KD", 0.34);
            this.exchangeValues.put("NZD", 1.63);
            break;
    case "New Zealand":
            this.exchangeValues.put("USD", 0.67);
            this.exchangeValues.put("EUR", 0.57);
            this.exchangeValues.put("GBP", 0.50);
            this.exchangeValues.put("PK", 111.30);
            this.exchangeValues.put("CNY", 4.60);
            this.exchangeValues.put("JPY", 71.26);
            this.exchangeValues.put("CAD", 0.88);
            this.exchangeValues.put("AUD", 0.92);
            this.exchangeValues.put("CHF", 0.61);
            this.exchangeValues.put("KD", 0.21);
            this.exchangeValues.put("NZD", 1.000);
            break;
```

```
                case "Kuwaiti Dinar":
                    this.exchangeValues.put("USD", 3.27);
                    this.exchangeValues.put("EUR", 2.76);
                    this.exchangeValues.put("GBP", 2.46);
                    this.exchangeValues.put("PK", 542.18);
                    this.exchangeValues.put("CNY", 22.36);
                    this.exchangeValues.put("JPY", 347.10);
                    this.exchangeValues.put("CAD", 4.27);
                    this.exchangeValues.put("AUD", 4.49);
                    this.exchangeValues.put("CHF", 2.98);
                    this.exchangeValues.put("NZD", 4.87);
                    this.exchangeValues.put("KD", 1.000);
                    break;
            }
        }

        // Initialize currencies Australian Dollar
        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("Canadian Dollar", "CAD") );
            currencies.add( new Currency("Australian Dollar", "AUD") );
            currencies.add( new Currency("New Zealand", "NZD") );
            currencies.add( new Currency("Swiss Franc", "CHF") );
            currencies.add( new Currency("Kuwaiti Dinar", "KD") );
            currencies.add( new Currency("British Pound", "GBP") );
            currencies.add( new Currency("Pakistan", "PK") );
```

```
		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;
	}
}
```

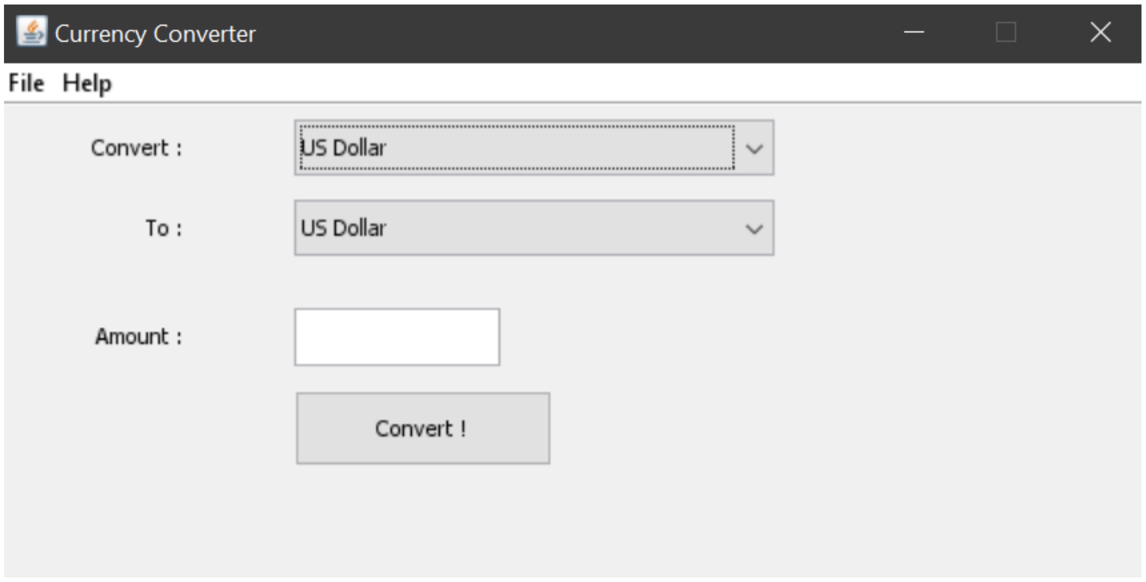
CurrencyConverter.java:

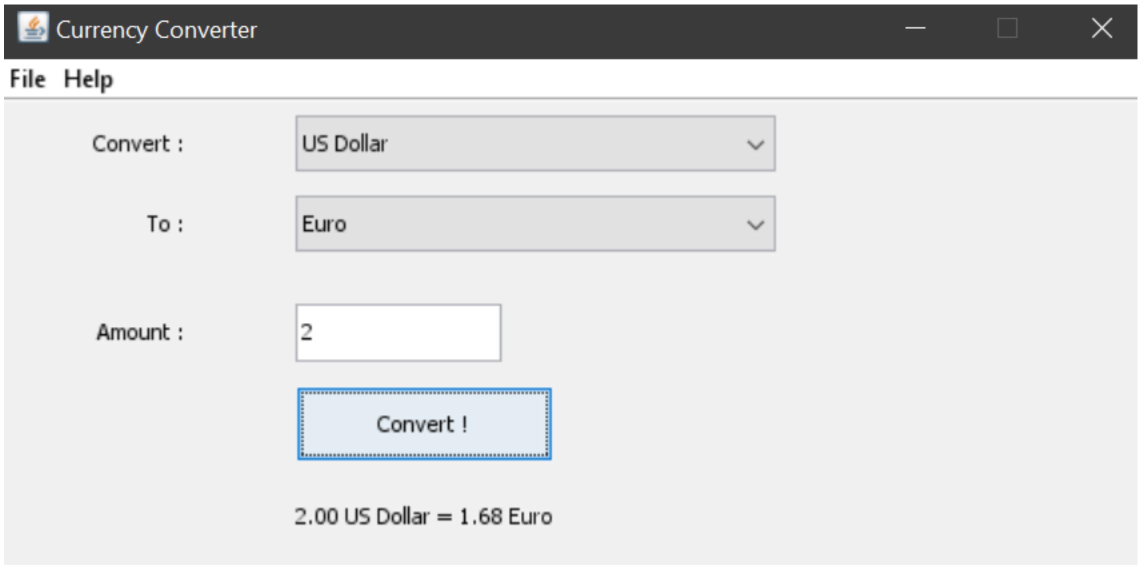
```
package currencyConverter;

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.java:

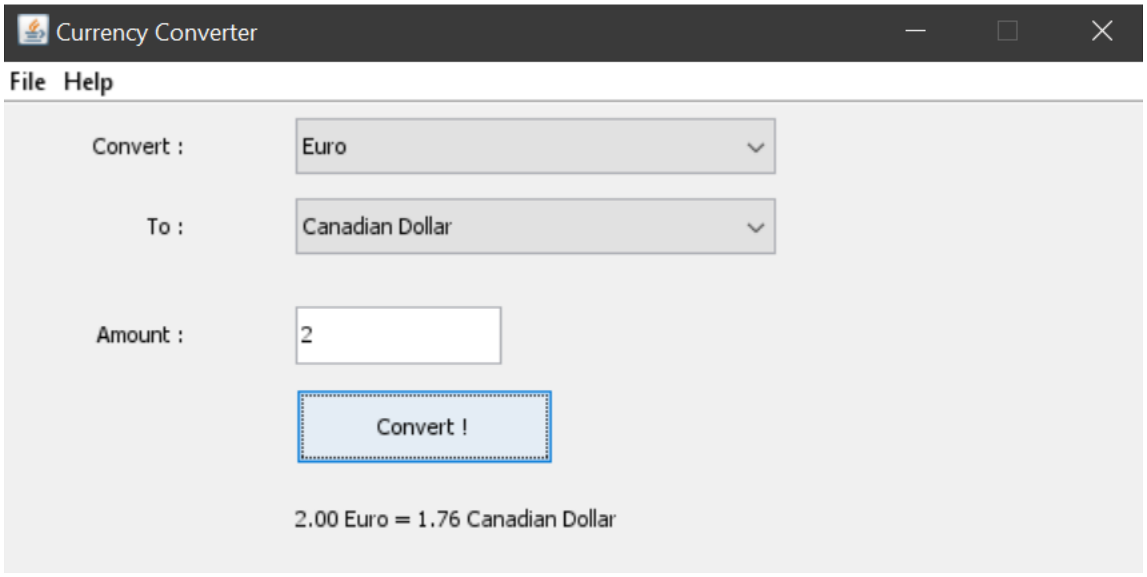
```
package currencyConverter;

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);
     }
   }
}
```

ScreenShots:



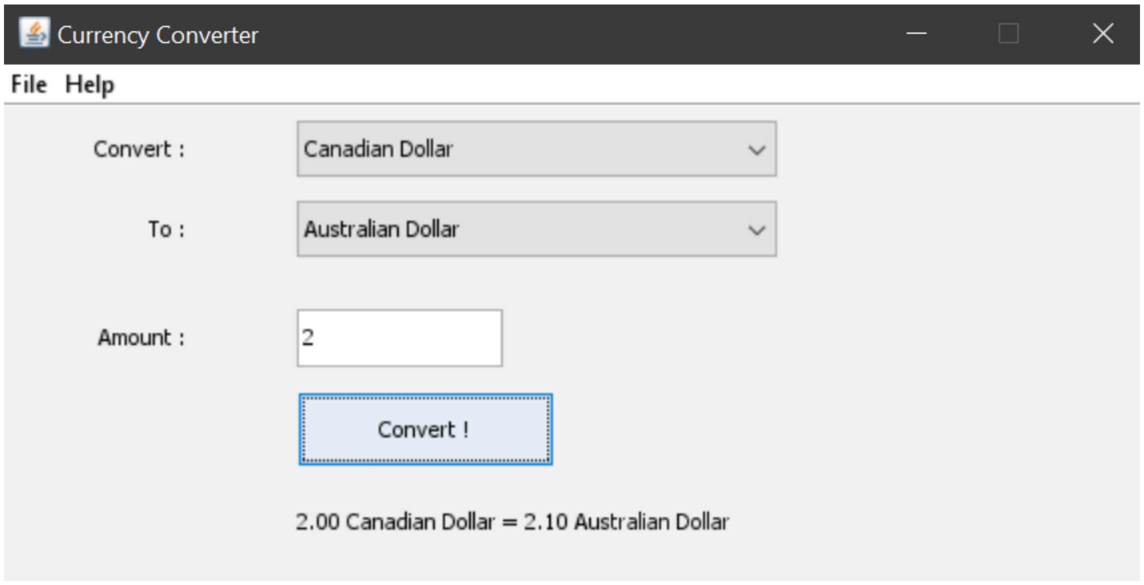


```
2.00 US Dollar = 1.68 Euro
```



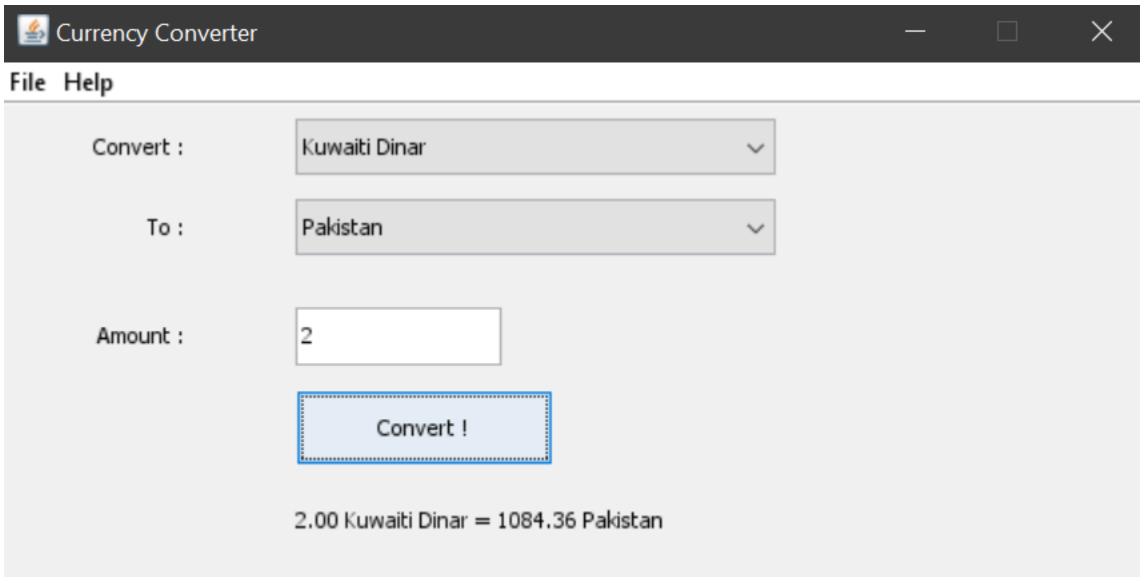
Convert : Euro
To : Canadian Dollar
Amount : 2

2.00 Euro = 1.76 Canadian Dollar



Convert : Canadian Dollar
To : Australian Dollar
Amount : 2

2.00 Canadian Dollar = 2.10 Australian Dollar



Convert : Kuwaiti Dinar

To : Pakistan

Amount : 2

```
2.00 Kuwaiti Dinar = 1084.36 Pakistan
```