Project Design

CMSC 495 6380 Current Trends and Projects in Computer Science (2218)

Sep 10 2021

Group 5

Danny Padro

Michael Schaffner

Shawn Kratochvil

Revision history

Revision number	Date	Description	Name
1	09/13/2021	Added a few amendments to the Pseudocode for all classes/subsystems section	Danny Padro
2	9/13/2021	Changed startup diagram creation steps for class and variables. Also changed the Shutdown diagram to destruction instead of lost messages.	Mike Schaffner
3	10/3/2021	Changed Event-trace diagram to same format as Startup/shutdown diagrams.	Mike Schaffner

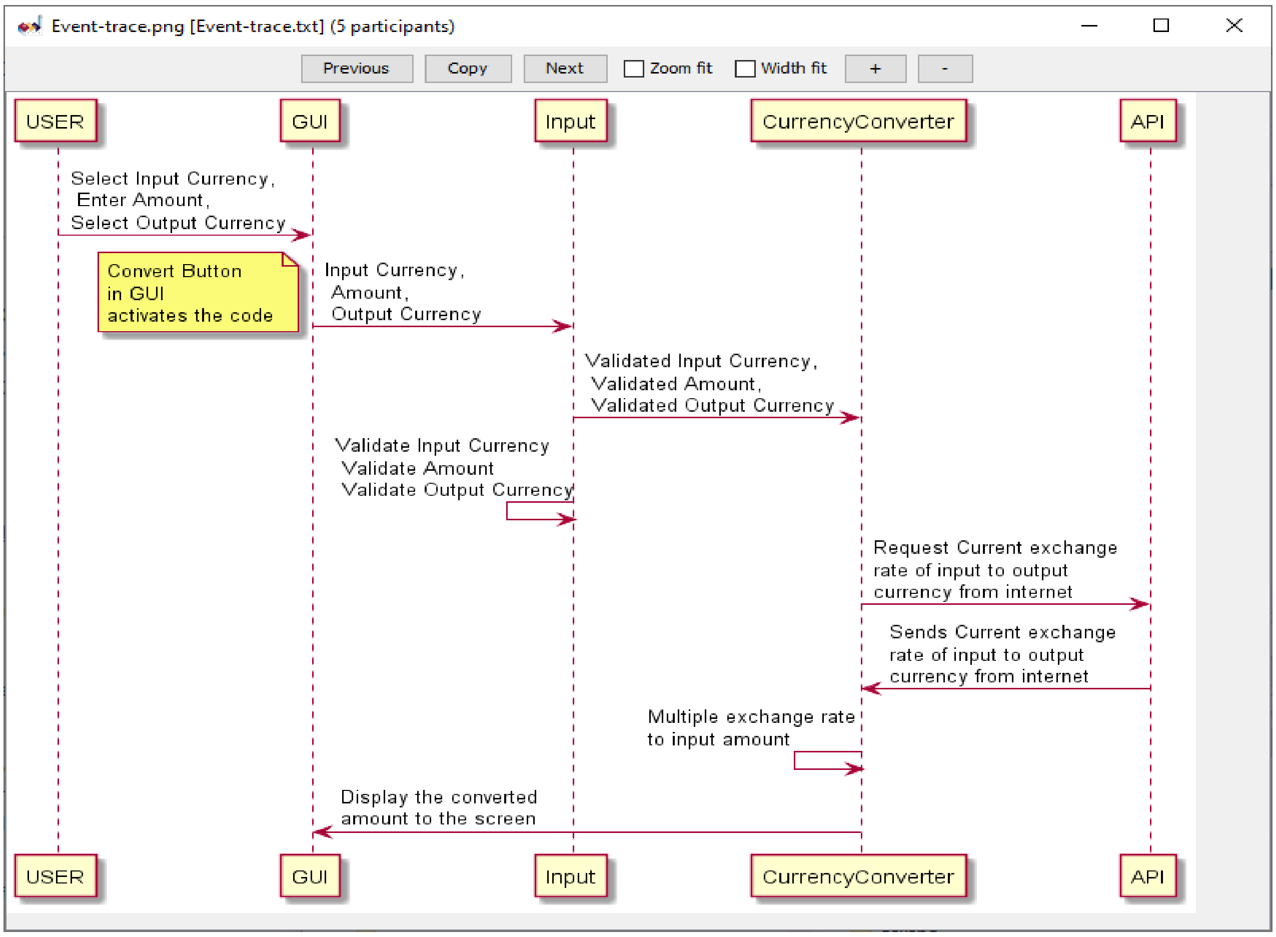
Event-trace diagram:

Scenario 1 (normal processing scenario):

Description: The user selects the input and output currency, then enters the amount to be converted, after which the program gets the current exchange rate via the API, multiplies that rate with the input amount, and displays it to the user.

Precondition: The application is running and connected to the internet.

Post-condition: The converted amount is displayed on the screen.


Event-trace.png [Event-trace.txt] (5 participants)
Previous
Copy
Next
Zoom fit
Width fit
+
-
USER
GUI
Input
CurrencyConverter
API
Select Input Currency, Enter Amount, Select Output Currency
Convert Button in GUI activates the code
Input Currency, Amount, Output Currency
Validated Input Currency, Validated Amount, Validated Output Currency
Validate Input Currency Validate Amount Validate Output Currency
Request Current exchange rate of input to output currency from internet
Sends Current exchange rate of input to output currency from internet
Multiple exchange rate to input amount
Display the converted amount to the screen
USER
GUI
Input
CurrencyConverter
API

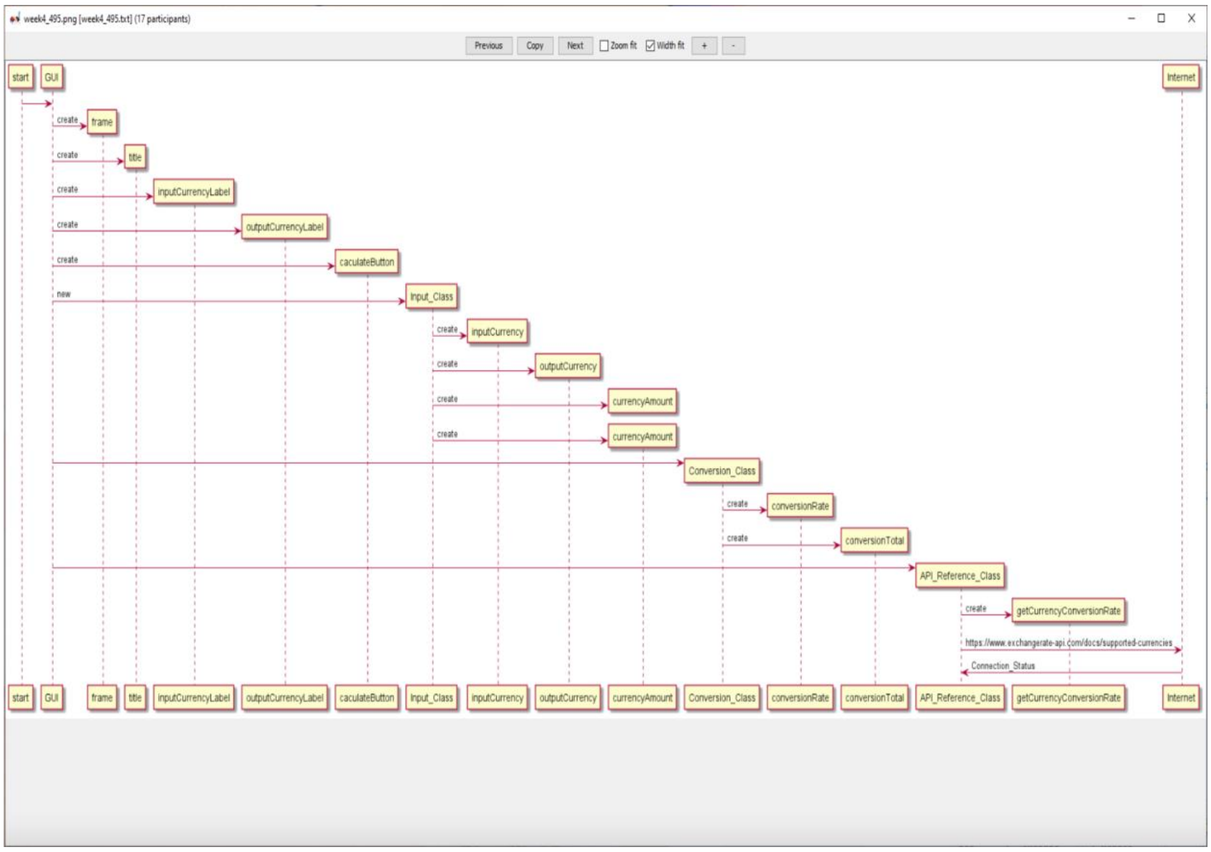
Startup Scenario:

Description: Upon a startup request the program creates the GUI, Input, Conversion, API Reference Classes. Each class initializes variables listed inside the class. The GUI Class initializes all class variables.

The INPUT Class initializes all class variables. No values are assigned at class creation. The Conversion Class initializes all class variables. No values are assigned at class creation.

The API Reference initializes all class variables and accesses the API for current conversion rates from the Internet.

Pre-condition: Program is not running



Post condition: Application is running and connected to the internet

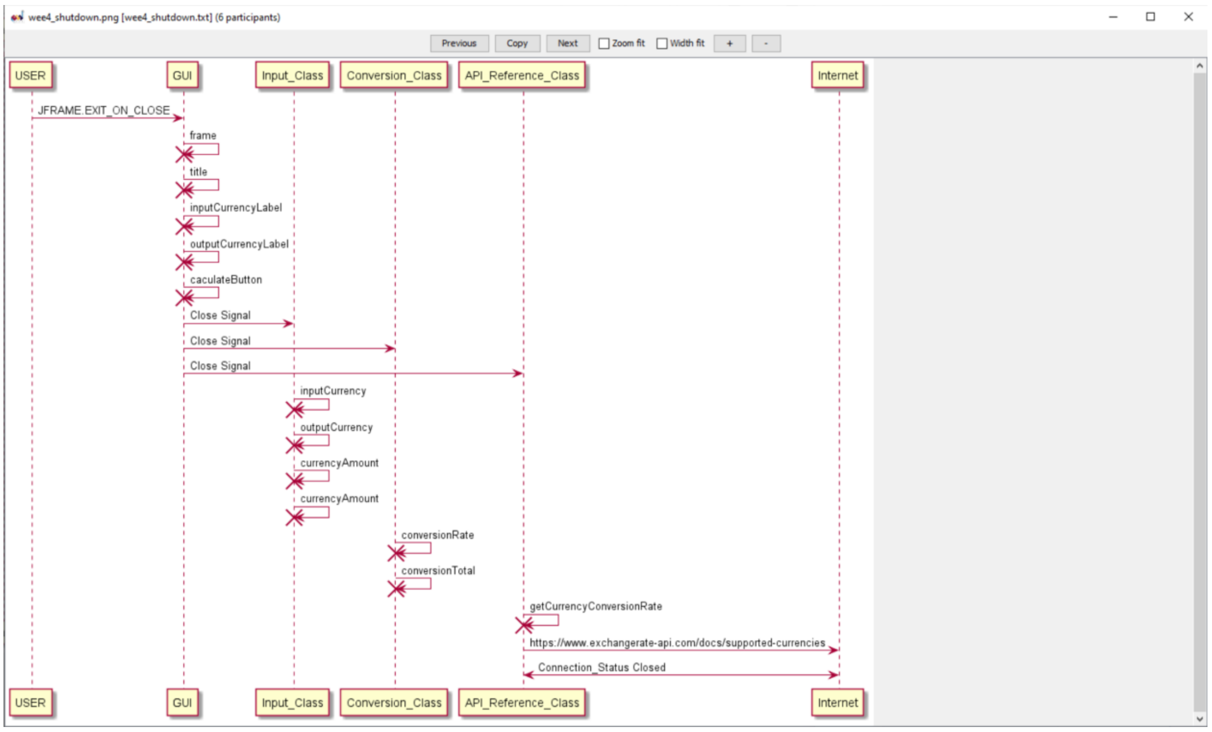
Shutdown Scenario:

Description: The user presses the close button to close the application

Pre-condition: Application is running and connected to the internet

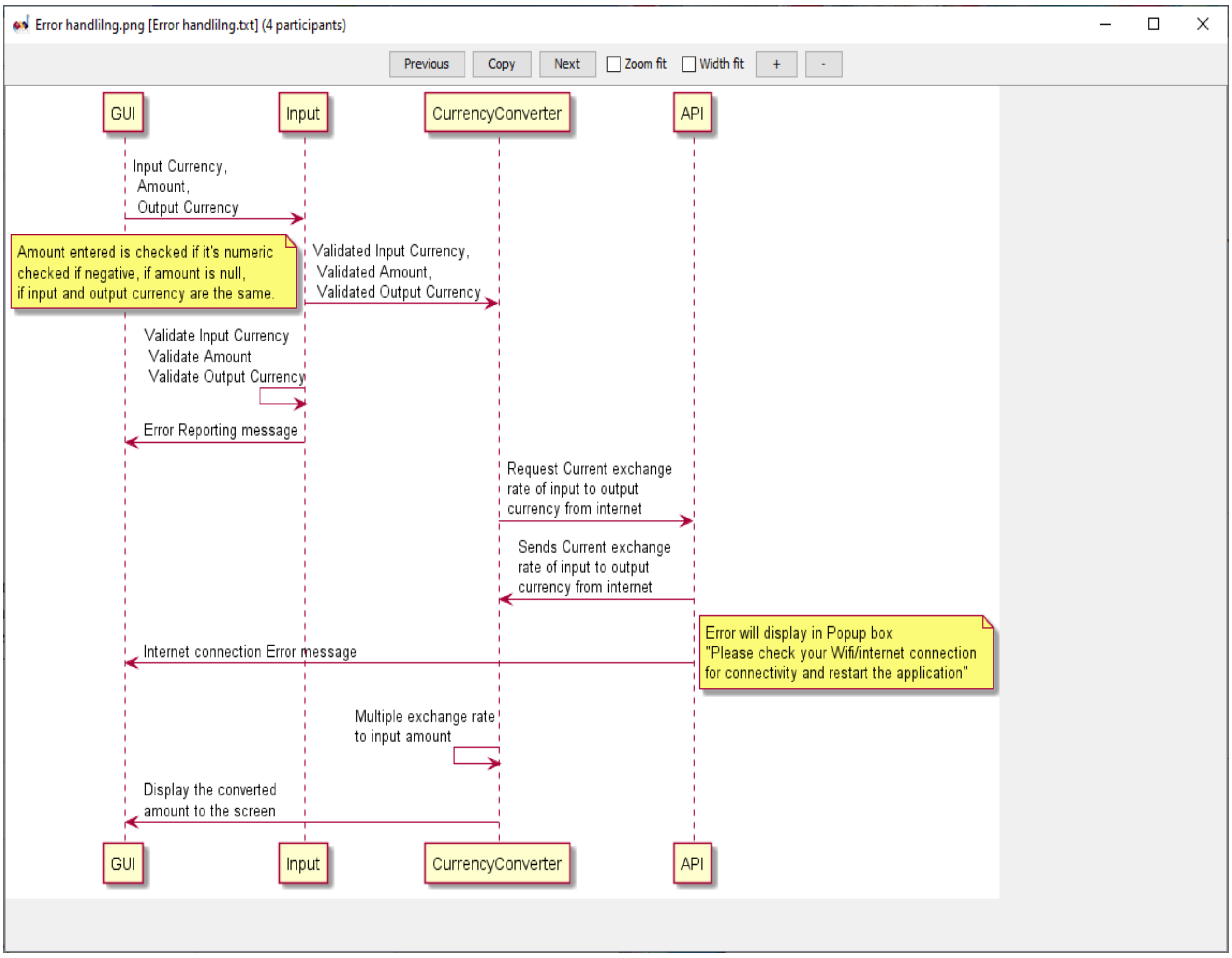
Post condition: Application is not running

Upon a shutdown request received from the User via the GUI Class the JFRAME.EXIT_ON_CLASE will be executed. This runs the standard Java close sequence destroying all variables created during startup and the program running state. The connection to the internet is closed. Program exits and GUI is destroyed.



Error-handling scenario:

If the user enters invalid numeric data for the currency Amount, the user will be propped to enter valid data until he/she enters valid data. The input and output currency will be handled by using JComboBoxes. As long as an input and output currency are both selected using these JComboBoxes, the program will allow for the user to press the calculate button (which will then determine if the input amount is valid).



Error handlilng.png [Error handlilng.txt] (4 participants)
Previous
Copy
Next
Zoom fit
Width fit
+
-
GUI
Input
CurrencyConverter
API
Input Currency, Amount, Output Currency
Amount entered is checked if it's numeric checked if negative, if amount is null, if input and output currency are the same.
Validated Input Currency, Validated Amount, Validated Output Currency
Validate Input Currency Validate Amount Validate Output Currency
Error Reporting message
Request Current exchange rate of input to output currency from internet
Sends Current exchange rate of input to output currency from internet
Error will display in Popup box "Please check your Wifi/internet connection for connectivity and restart the application"
Internet connection Error message
Multiple exchange rate to input amount
Display the converted amount to the screen
GUI
Input
CurrencyConverter
API

Class Design:

a. Input subsystem:

class Input{

String currencyAmount;

Double currencyAmountIfValid;

String inputCurrency;

String outputCurrency;

<u>Helper methods:</u>
Boolean currencyAmountIsValid(); // checks whether the string the user enters for the currency is a valid double

Double calculateConversion(double currencyAmount); //determines the output Currency amount using the conversion rate provided by the API, displays it to the GUI

Boolean inputCurrencyIsSelected(); //determines if the inputCurrency is selected from the GUI's dropdown list
Boolean outputCurrencyIsSelected(); //determines if the outputCurrency is selected from the GUI's dropdown list

end helper methods

NOTE: Rather than accepting string input for the inputCurrency and outputCurrency, we may use a dropdown menu of the available currencies that the user can select on the GUI. This makes it so the user does not have to type anything (and thus, there's no error handling necessary at this point because the program doesn't have to check if the currency string they entered is valid or not) and instead has to select from a list.

While (inputCurrencyIsSelected() and outputCurrencyIsSelected()) //waits for input to be entered by user

{

If (userCalculateButtonIsPressed()){

```
If(currencyAmountIsValid(currencyAmount)){

calculateConversion(currencyAmountIfValid);
}

else{

 Set errorField("Input String cannot be parsed to Double.");

Set errorField2(“Please re-enter input data.”);

currencyAmountField.clear();

}

}

Boolean currencyAmountIsValid(String currencyAmount){

Try { currencyAmountIfValid =  Double.parseDouble(currencyAmount);

Return true;} catch (NumberFormatException e) {

    }

Return false;}

}
```

NOTE: I have not yet looked at the API so the pseudocode for this will be rather rudimentary.

b. Conversion subsystem:

```
class Conversion{
double conversionRate;

double conversionTotal;

Double getConversionRate (String inputCurrency, String outputCurrency){

try{
if (cannotAccessDataBase()){

throw new Exception();}

return conversionRate; // The API will need the inputCurrency and outputCurrency
to determine the appropriate conversion rate between the two currencies
}}

Double calculateConversion(double currencyAmount){
//determines what inputCurrency is in outputCurrency metrics

 conversionTotal= conversionRate * currencyAmount;

return conversionTotal;

}

}
```

c. API Reference subsystem:

```
class API Reference

{

double getCurrencyConversionRate throws Exception{

try{ if (cannotAccessDataBase()){

throw new Exception();

}
return conversionRate;

}
//accesses the conversion rate from the API's database, if cannot access, an exception is thrown

catch (Exception e){

e.printStackTrace();

}


}

}
```

d. GUI subsystem:

```
class GUI //this will set up the basic structure of the GUI with buttons, combo boxes, labels
```

```
{


JFrame frame = new JFrame("Currency Converter");

JLabel title = new JLabel(“Currency Converter”);

JLabel inputCurrencyLabel = new JLabel(“Input Currency”);

JLabel outputCurrencyLabel = new JLabel(“Output Currency”);

JLabel errorField = new JLabel(“”); //will be used to show erros/exceptions
JLabel errorField2 = new JLabel(“”);

 frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

 frame.setSize(1420,1080);

  JButton calculateButton = new JButton("Calculate"); //when pressed, this will
display the conversionTotal to the JTextField of the same name

JComboBox inputCurrency = new JComboBox(“”); //allows the user to select the
input currency to be converted

JComboBox outputCurrency = new JComboBox(“”); //allows the user to select the
output currency

JTextField conversionTotalField = new JTextField(“”); //will display the result of
calculateConversion()

JTextField currenyAmountField = new JTextField(“”); //will be an empty
JTextField which will accept the user input for the amount of input currency

Frame.getContentPane().add(currencyAmountField);
frame.getContentPane().add(inputCurrencyLabel);
frame.getContentPane().add(inputCurrency);

frame.getContentPane().add(outputCurrencyLabel);

frame.getContentPane().add(outputCurrency);

frame.getContentPane().add(conversionTotalField);
frame.getContentPane().add(calculateButton);

      frame.setVisible(true);

}
```

Unresolved risks and risk mitigation:

One unresolved risk is the lack of internet connection. The currency convertor will not update current currency values without an internet connection. The project has not included the database storage of currency values to function without the internet connection.