

Project 4 Task 2 - Currency Conversion Application

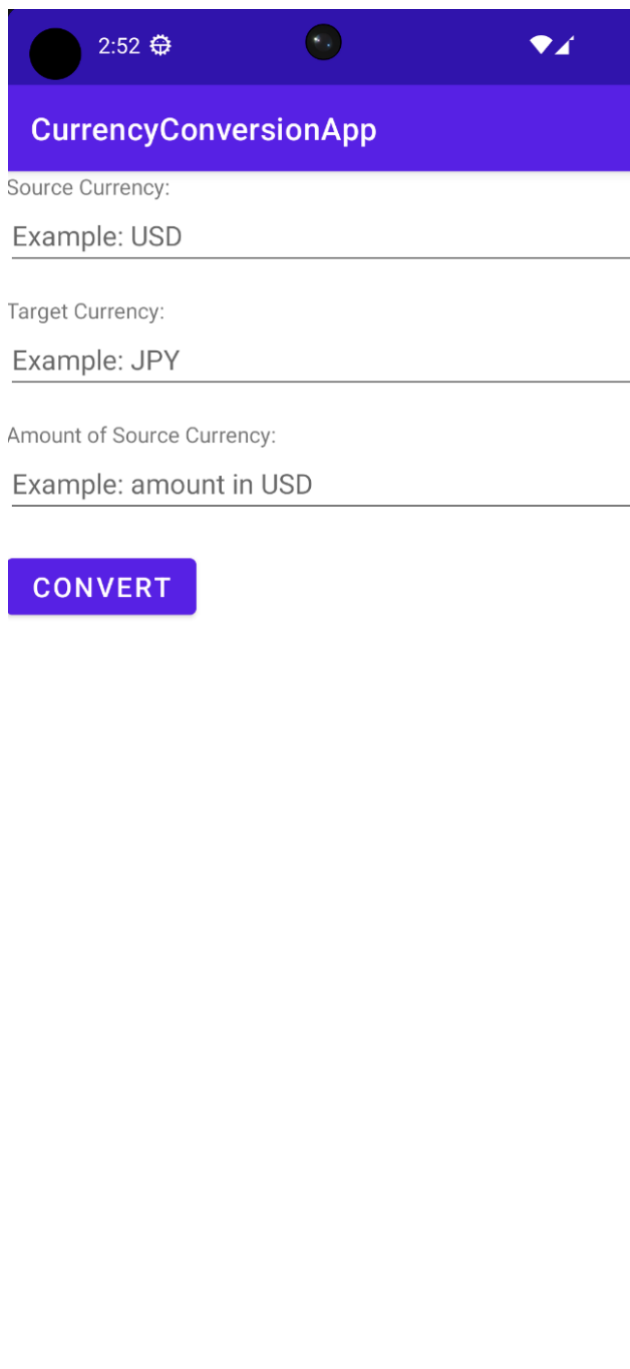
by Kathy Chiang (AndrewID: pohsingc)

Description:

1. Implement a native Android application

- a. My application uses TextView, EditText, and Button. See content_main.xml for details of how they are incorporated into the LinearLayout.

Here is a screenshot of the layout before the information has been fetched:



- b. Requires input from the user

Here is a screenshot of the user searching for the conversion from 45,000 Japanese Yen to US Dollar:

The screenshot shows a mobile application interface for currency conversion. At the top, the status bar displays the time 2:53, a gear icon, and signal strength indicators. The app title "CurrencyConversionApp" is centered in a purple header. Below the header, the form is divided into three sections: "Source Currency:" with "JPY" entered, "Target Currency:" with "USD" entered, and "Amount of Source Currency:" with "45000" entered. A purple "CONVERT" button is positioned below the amount field. At the bottom, a virtual keyboard is visible, with a URL bar showing "https://kathyc...".

- c. Makes an HTTP request (using an appropriate HTTP method) to my web service

My application does an HTTP GET request in GetRate.java. The HTTP request is: <https://kathycceffective-orbit-xxv4xw7pw4c6494-8080.preview.app.github.dev/currencies?from=sourceCurrency&totargetCurr>

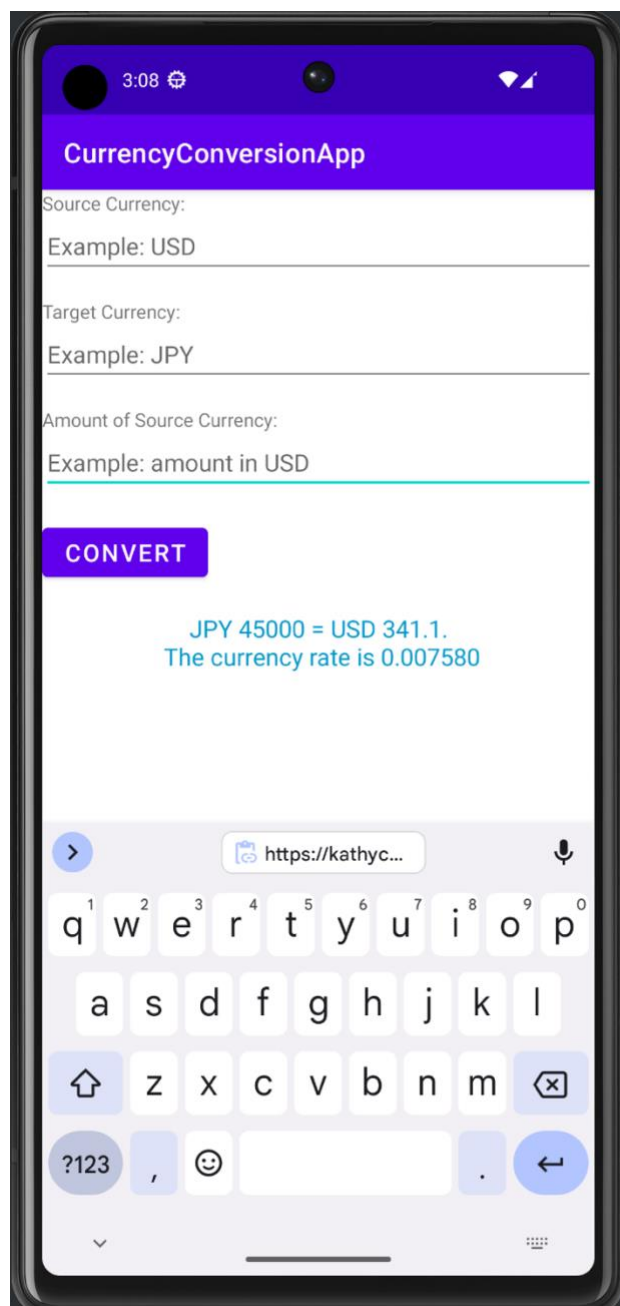
[ency&amount0givenAmount](#), where sourceCurrency is the currency that would be converted, targetCurrency is the converted currency, and the givenAmount is the amount of source currency.

The search method makes this request of my web application, parse the returned JSON to get the converted amount of target currency and the conversion rate between two currencies.

- d. Receives and parses an JSON formatted reply from the web service
An example of the JSON reply is:

```
{"result":"success","rate":"0.007580","convertedAmount":"341.1","time_last_update":"Sat, 08 Apr 2023 00:00:02 +0000"}
```

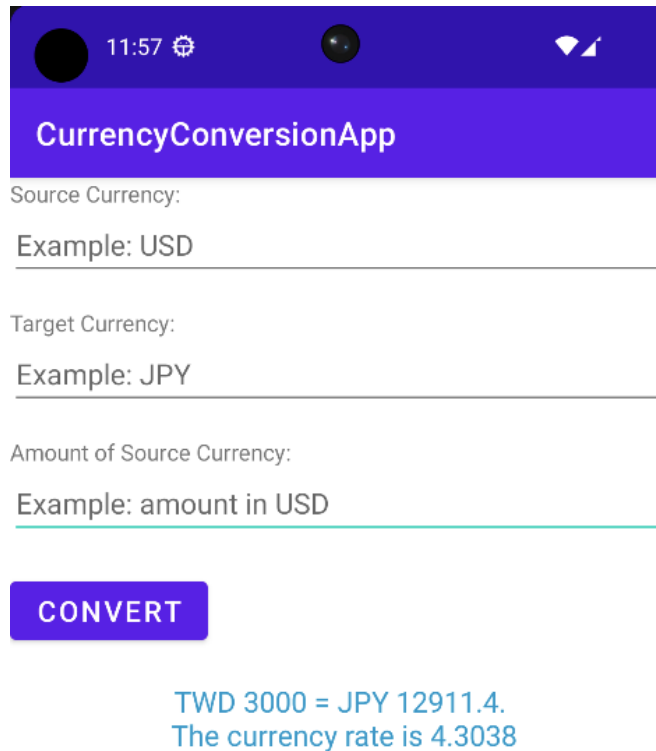
- e. Displays the conversion results to the user
Here is the screenshot after the conversion results have been returned:



- f. Is repeatable (I.e the user can repeatedly reuse the application without restarting it)

The user can type in another search terms and hit CONVERT.

Here is an example of converting TWD to JPY:



11:57

CurrencyConversionApp

Source Currency:

Example: USD

Target Currency:

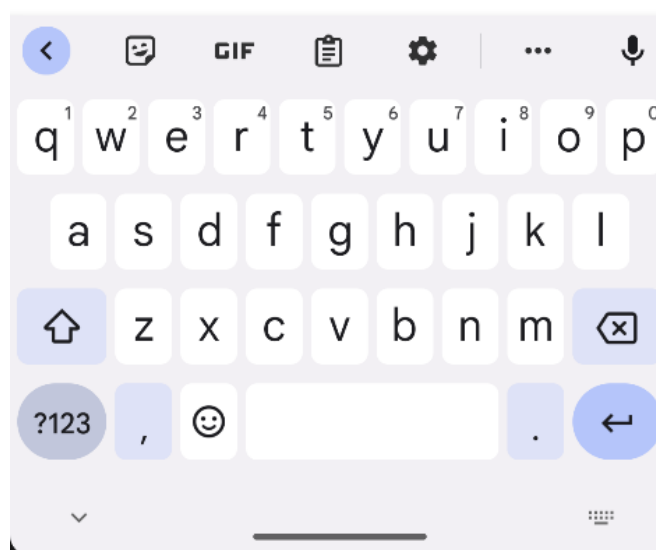
Example: JPY

Amount of Source Currency:

Example: amount in USD

CONVERT

TWD 3000 = JPY 12911.4.
The currency rate is 4.3038



2. Implement a web application, deployed to GitHub CodeSpaces

The URL of my web service deployed to GitHub CodeSpaces is <https://kathyccc-miniature-robot-67r4wppg93jrj-8080.preview.app.github.dev/>

The project directory name project-4-task-2-Kathyccc.

- a. Using an HttpServlet to implement a simple API
In my web app project:
Model: CurrencyModel.java
Controller: CurrencyServlet.java
View: dashboard.jsp (for dashboard)
- b. Receives an HTTP request from the native Android application
CurrencyServlet.java receives the HTTP GET request with the arguments "from", "to" and "amount". It passes this search string on to the model.
- c. Executes business logic appropriate to the web application
CurrencyModel.java makes an HTTP request to: <https://v6.exchangerate-api.com/v6/>

It then parses the JSON response and extracts the parts it needs to response to the Android application.
- d. Replies to the Android application with an JSOB formatted response.
Currency.jsp formats the response to the mobile application in a simple JSON format of my own design:

```
<%@ page import="com.google.gson.JsonObject" %>
<%@ page import="com.google.gson.Gson" %>
<%
    // Get the JSON response from the servlet
    String jsonString = (String) request.getAttribute("response");

    Gson gson = new Gson();
    JsonObject jsonResponse = gson.fromJson(jsonString, JsonObject.class);

    // Get the relevant fields from the JSON object
    String result = jsonResponse.get("result").getAsString();
    String rate = jsonResponse.get("rate").getAsString();
    String convertedAmount =
jsonResponse.get("convertedAmount").getAsString();
    String time_last_update =
jsonResponse.get("time_last_update").getAsString();

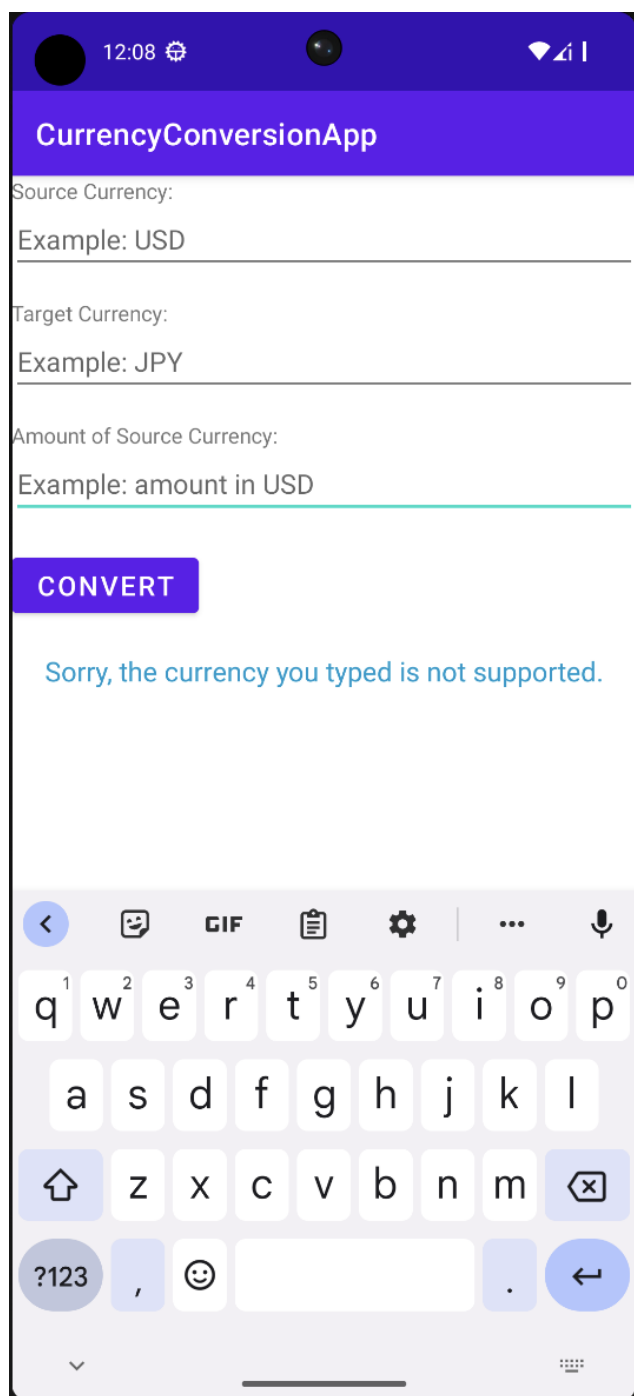
    // Create a JSON object with the relevant fields
    JsonObject res = new JsonObject();
    res.addProperty("result", result);
    res.addProperty("rate", rate);
    res.addProperty("convertedAmount", convertedAmount);
    res.addProperty("time_last_update", time_last_update);

    // Send the JSON response to the Android app
    out.print(res.toString());
%>
```

3. Handle error conditions

```
// If the input currency is not supported
} else {
    // Otherwise, set the feedback text to display an error message
    feedbackView.setText("Sorry, the currency you typed is not supported.");
    System.out.println("currency not supported");

    // Make the feedback view visible
    feedbackView.setVisibility(View.VISIBLE);
}
```



4. Log useful Information

Store useful information into database, such as the search time, rate, and converted amount.

```
// Add the fields to the document
document.append("Result", result);
document.append("Search Time", new Date());
document.append("Last Update Time", time_last_update);
document.append("from", from);
document.append("to", to);
document.append("rate", rate);
document.append("givenAmount", givenAmount);
document.append("convertedAmount", convertedAmount);
document.append("responseTime", responseTime);

// Write the data to the database
model.writeDB(document, collectionName: "currencySearch");
```

5. Store the log information in MongoDB

```
/**
 * Writes a document to a specified MongoDB collection.
 *
 * @param document the document to write to the collection
 * @param collectionName the name of the collection to write the document to
 */
2 usages
public void writeDB(Document document, String collectionName){
    // Create a new MongoDB client and connect to the database
    // Reference: MongoDB
    ConnectionString connectionString = new ConnectionString("mongodb://kathyc:May
    MongoClientSettings settings = MongoClientSettings.builder()
        .applyConnectionString(connectionString)
        .serverApi(ServerApi.builder()
            .version(ServerApiVersion.V1)
            .build())
        .build();

    MongoClient mongoClient = MongoClient.create(settings);
    MongoDB database = mongoClient.getDatabase(s: "currencies");
    MongoCollection<Document> collection = database.getCollection(collectionName);

    // Insert the Document into the MongoDB collection
    collection.insertOne(document);

    mongoClient.close();
}
```

6. Display operations analytics and full logs on a web-based dashboard

Please access the dashboard using this URL: <https://kathyccc-miniature-robot-67r4wppg93jrj-8080.preview.app.github.dev/dashboard>

Logs

| Search Time | Last Update Time | From | To | Rate | Given Amount | Converted Amount | Error Message |
|------------------------------|---------------------------------|------|-----|----------|--------------|------------------|---------------|
| null | Tue, 04 Apr 2023 00:00:02 +0000 | USD | JPY | 132.4575 | null | 26491.5 | |
| Wed Apr 05 18:52:26 UTC 2023 | Wed, 05 Apr 2023 00:00:02 +0000 | USD | JPY | 131.7266 | null | 13172.66 | |
| Wed Apr 05 19:02:13 UTC 2023 | Wed, 05 Apr 2023 00:00:02 +0000 | USD | JPY | 131.7266 | null | 13172.66 | |
| Wed Apr 05 22:05:15 UTC 2023 | Wed, 05 Apr 2023 00:00:02 +0000 | USD | JPY | 131.7266 | null | 13172.66 | |
| Wed Apr 05 22:07:02 UTC 2023 | Wed, 05 Apr 2023 00:00:02 +0000 | USD | JPY | 131.7266 | null | 13172.66 | |
| Fri Apr 07 17:17:40 UTC 2023 | Fri, 07 Apr 2023 00:00:01 +0000 | USD | JPY | 131.5888 | 200 | 26317.76 | |
| Fri Apr 07 17:51:29 UTC 2023 | Fri, 07 Apr 2023 00:00:01 +0000 | USD | JPY | 131.5888 | 200 | 26317.76 | |
| Fri Apr 07 18:05:44 UTC 2023 | Fri, 07 Apr 2023 00:00:01 +0000 | JPY | USD | 0.007599 | 250 | 1.89975 | |
| Fri Apr 07 19:10:50 UTC 2023 | Fri, 07 Apr 2023 00:00:01 +0000 | USD | JPY | 131.5888 | 1000 | 131588.8 | |
| Fri Apr 07 19:11:09 UTC 2023 | Fri, 07 Apr 2023 00:00:01 +0000 | USD | JPY | 131.5888 | 1000 | 131588.8 | |
| Fri Apr 07 20:04:26 UTC 2023 | Fri, 07 Apr 2023 00:00:01 +0000 | USD | JPY | 131.5888 | 200 | 26317.76 | |
| Fri Apr 07 20:04:50 UTC 2023 | Fri, 07 Apr 2023 00:00:01 +0000 | USD | JPY | 131.5888 | 200 | 26317.76 | |
| Fri Apr 07 20:11:27 UTC 2023 | Fri, 07 Apr 2023 00:00:01 +0000 | USD | JPY | 131.5888 | 1000 | 131588.8 | |
| Fri Apr 07 20:38:49 UTC 2023 | Fri, 07 Apr 2023 00:00:01 +0000 | JPY | USD | 0.007599 | 5000 | 37.995 | |
| Fri Apr 07 21:39:10 UTC 2023 | Fri, 07 Apr 2023 00:00:01 +0000 | USD | JPY | 131.5888 | 620 | 81585.056 | |
| Fri Apr 07 22:46:55 UTC 2023 | Fri, 07 Apr 2023 00:00:01 +0000 | JPY | USD | 0.007599 | 30000 | 227.97 | |
| Sat Apr 08 01:18:56 UTC 2023 | Sat, 08 Apr 2023 00:00:02 +0000 | JPY | USD | 0.007580 | 25000 | 189.5 | |
| Sat Apr 08 01:33:54 UTC 2023 | Sat, 08 Apr 2023 00:00:02 +0000 | JPY | USD | 0.007580 | 2400 | 18.192 | |
| Sat Apr 08 01:34:43 UTC 2023 | Sat, 08 Apr 2023 00:00:02 +0000 | JPY | USD | 0.007580 | 2400 | 18.192 | |
| Sat Apr 08 01:38:54 UTC 2023 | Sat, 08 Apr 2023 00:00:02 +0000 | USD | JPY | 131.9339 | 250 | 32983.475 | |
| Sat Apr 08 01:39:26 UTC 2023 | Sat, 08 Apr 2023 00:00:02 +0000 | USD | JPY | 131.9339 | 250 | 32983.475 | |
| Sat Apr 08 01:52:06 UTC 2023 | Sat, 08 Apr 2023 00:00:02 +0000 | JPY | USD | 0.007580 | 50000 | 379 | |
| Sat Apr 08 01:55:28 UTC 2023 | Sat, 08 Apr 2023 00:00:02 +0000 | USD | JPY | 131.9339 | 900 | 118740.51 | |
| Sat Apr 08 02:05:54 UTC 2023 | Sat, 08 Apr 2023 00:00:02 +0000 | USD | JPY | 131.9339 | 600 | 79160.34 | |
| Sat Apr 08 02:06:37 UTC 2023 | Sat, 08 Apr 2023 00:00:02 +0000 | USD | JPY | 131.9339 | 600 | 79160.34 | |

Popular Currencies

1. USD
2. JPY
3. TWD
4. DKK
5. AMD

Average Latency Time

The average latency time to retrieve conversion results and generating the response is 315 milliseconds

Total Number of Search

The total number of search for currency conversion is 58.

7. Here is the video that shows how my Android application works:

<https://youtu.be/JeOMnpKdTxS>