

Project 4 Task 1 – Currency Exchange Rate App, by Xian Zhang

Description:

My application takes selected radio box from the user, and uses it to fetch and display currency exchange rate (base currency is EUR).

Here is how my application meets the task requirements

1. Implement a native Android application

The name of my native Android application project in Android Studio is: Project4Android

1.1. Has at least three different kinds of views in your Layout (TextView, EditText, ImageView, etc.)

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

Here is a screenshot of the layout before the currency exchange rate has been fetched.

A screenshot of a mobile application interface. At the top, there is a blue status bar with icons for settings, a circular progress indicator, a document icon, and the time 4:20. Below the status bar, the text "Please choose a currency below:" is displayed. Underneath this text is a grey button labeled "SUBMIT". Below the button is a list of five radio button options: "USD", "INR", "AUD", "CAD", and "CNY". The "INR" option is selected, indicated by a filled radio button.

1.2. Requires input from the user

Here is a screenshot of the user searching for INR currency exchange rate (base currency is EUR)

The screenshot shows a mobile application interface. At the top, there is a status bar with icons for settings, a circular indicator, a document icon, and the time 4:23. Below the status bar, the text "Please choose a currency below:" is displayed. Underneath this text is a grey button labeled "SUBMIT". Below the button, there are five radio button options: "USD", "INR", "AUD", "CAD", and "CNY". The "INR" option is selected, indicated by a pink dot next to the radio button.

1.3. Makes an HTTP request (using an appropriate HTTP method) to your web service

My application does an HTTP GET request in `Project4task1.java`. The HTTP request is: "
`https://arcane-meadow-96091.herokuapp.com/WebServer?key=" +`
`searchTerm` where `search` is the user's search term.

The `search` method makes this request of my web

application, parses the returned JSON to find the fetches currency exchange rate and returns number back to user.

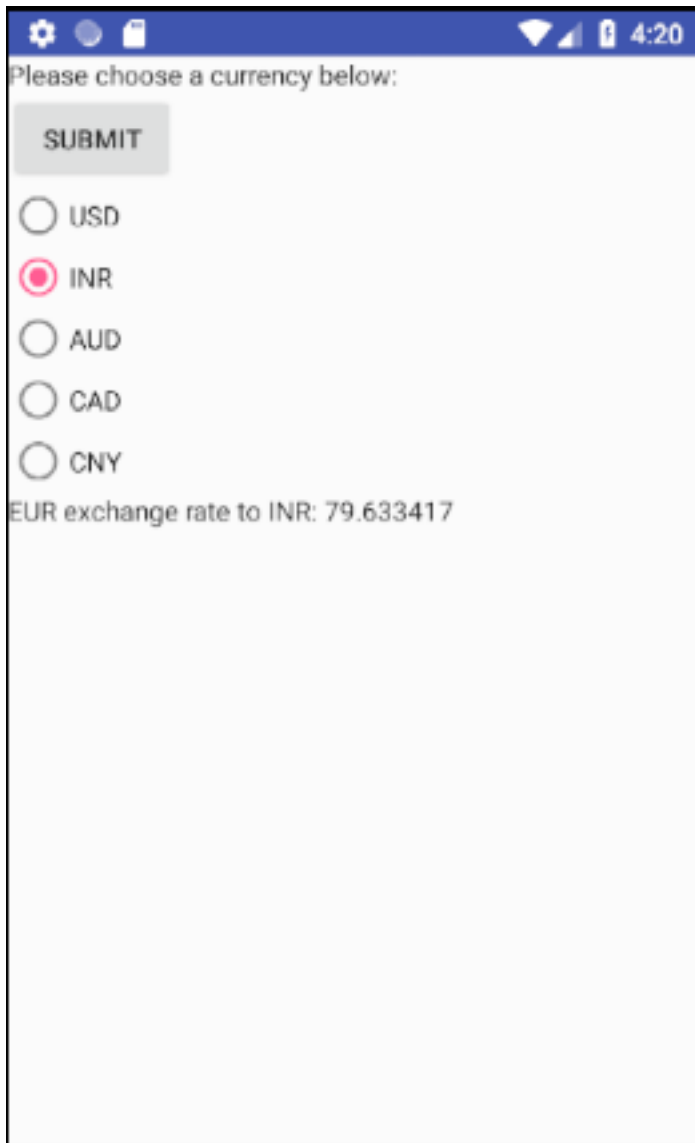
1.4. Receives and parses an XML or JSON formatted reply from the web service

An example of the JSON reply is:

```
{"success":true,"timestamp":1522986243,"base":"EUR","date":"2018-04-06","rates":{"INR":79.633417}}
```

1.5. Displays new information to the user

Here is the screen shot after the INR currency exchange rate (base currency is EUR) has been returned.



A screenshot of a mobile application interface. At the top is a blue status bar with icons for settings, a blue circle, a document, Wi-Fi, cellular signal, battery, and the time 4:20. Below the status bar, the text "Please choose a currency below:" is displayed. Underneath is a grey button labeled "SUBMIT". Below the button are five radio button options: "USD", "INR" (which is selected with a pink dot), "AUD", "CAD", and "CNY". At the bottom of the screen, the text "EUR exchange rate to INR: 79.633417" is shown.

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

The user can select in another search currency and hit Submit. Here is an example of having typed in "CNY".

The screenshot shows a mobile web application interface. At the top, there is a status bar with icons for settings, a blue circle, a document, and the time 4:33. Below the status bar, the text "Please choose a currency below:" is displayed. A "SUBMIT" button is located below the text. Below the button, there are five radio button options: "USD", "INR", "AUD", "CAD", and "CNY". The "CNY" option is selected, indicated by a red dot. Below the radio buttons, the text "EUR exchange rate to CNY: 7.72407" is displayed.

2. Implement a web application, deployed to Heroku

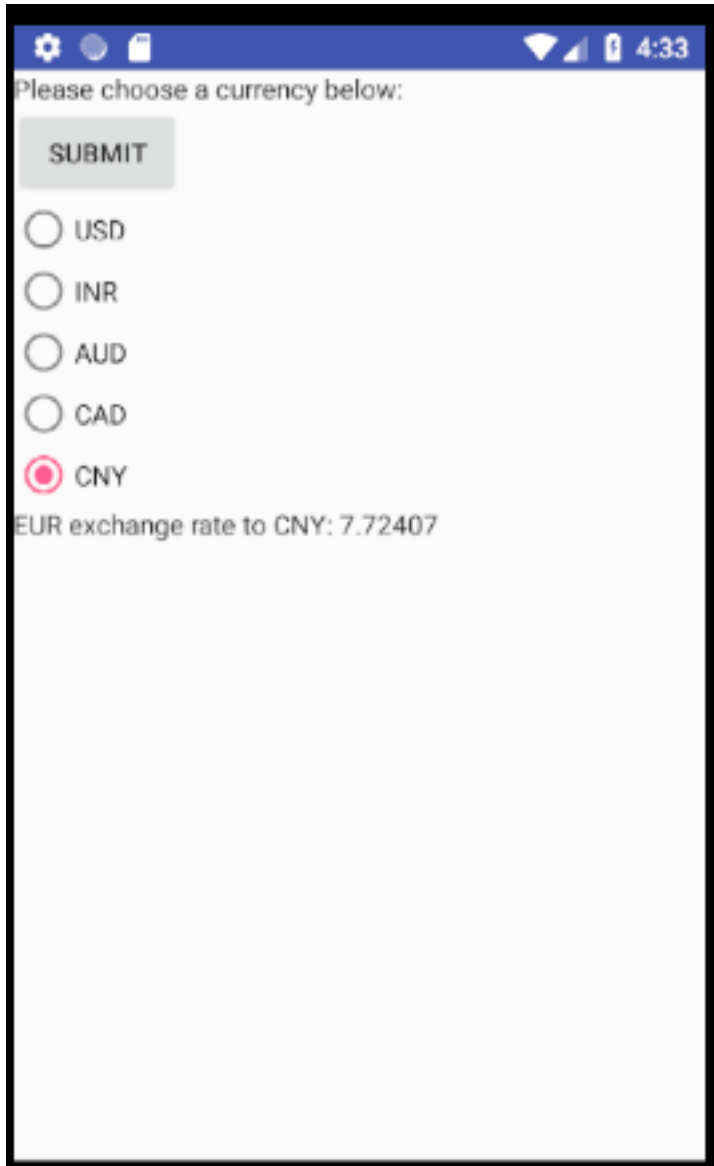
The URL of my web service deployed to Heroku is <https://arcane-meadow-96091.herokuapp.com>. The project directory name is Project4Task1.

2.1. Using an HttpServlet to implement a simple (can be a single path) API

In my web app project:

Model: Project4Task.java

Controller: Controller.java



2.2. Receives an HTTP request from the native Android application

Project4Task1.java receives the HTTP GET request with the argument "search". It passes this search string on to the model.

2.3. Executes business logic appropriate to your application

Project4Task1.java makes an HTTP request to:
`http://data.fixer.io/api/latest?access_key=17ad1c66c2e75424e7d84144f747de61&symbols="+ search + "&format=2.`

It then parses the JSON response and extracts the parts it needs to respond to the Android application.

2.4. Replies to the Android application with an XML or JSON formatted response.

Use `StringBuilder` formats the response to the mobile application in a simple JSON format of my own design:

```
{"success":true,"timestamp":1523032444,"base":"EUR","date":"2018-04-06","rates":{"CNY":7.737323}}
```