Mobile challenge for Bemobile.

International Business Men.

You work for GNB (Goliath National Bank), and your manager, Barney Stinson, has asked you to design and implement a mobile application to help the firm executives who are always flying around the globe. Your executives need a list of every product GNB trades with, and the total sum of sales of those products in different currencies.

For this task you have to use a web service. This web service can provide its results both in XML or in JSON format. You are free to choose the one you feel more comfortable with (you only need to use one of the formats). To use the XML format use the .xml extension; to use the JSON format use the .json extension. The web service provides you with two different sets of data:

- http://quiet-stone-2094.herokuapp.com/rates.xml or http://quiet-stone-2094.herokuapp.com/rates.json will return you a document with the following formats:

XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<rates>
  <rate from="EUR" to="USD" rate="1.359"/>
  <rate from="CAD" to="EUR" rate="0.732"/>
  <rate from="USD" to="EUR" rate="0.736"/>
  <rate from="EUR" to="CAD" rate="1.366"/>
</rates>
```

JSON:

```
[
  { "from": "EUR", "to": "USD", "rate": "1.359" },
  { "from": "CAD", "to": "EUR", "rate": "0.732" },
  { "from": "USD", "to": "EUR", "rate": "0.736" },
  { "from": "EUR", "to": "CAD", "rate": "1.366" }
]
```

Each dictionary from the array specifies the conversion rate from one currency to another (when the direct conversion is given, the reverse one is also provided), but some conversions may not be specified, and in case they are needed, they will have to be calculated using the already known conversions. For example, in the sample data we don't provide the USD to CAD conversion, that should be calculated from the USD to EUR and the EUR to CAD conversions.

- http://quiet-stone-2094.herokuapp.com/transactions.xml or http://quiet-stone-2094.herokuapp.com/transactions.json will return you a document with the following formats:

XML:

```
<?xml version="1.0" encoding="UTF-8"?> <transactions>
  <transaction sku="T2006" amount="10.00" currency="USD"/>
  <transaction sku="M2007" amount="34.57" currency="CAD"/>
 <transaction sku="R2008" amount="17.95" currency="USD"/>
 <transaction sku="T2006" amount="7.63" currency="EUR"/>
 <transaction sku="B2009" amount="21.23" currency="USD"/>
 ...
</transactions>
```

JSON:

```
[
 { "sku": "T2006", "amount": "10.00", "currency": "USD" },
 { "sku": "M2007", "amount": "34.57", "currency": "CAD" },
 { "sku": "R2008", "amount": "17.95", "currency": "USD" },
 { "sku": "T2006", "amount": "7.63", "currency": "EUR" },
 { "sku": "B2009", "amount": "21.23", "currency": "USD" }
]
```

Each dictionary represents a transaction of a given product (indicated by the product SKU) in given currency for the given amount.

Your application should download this information upon starting and give the user the choice of which product they want to see. When the user selects a product, the application must show each of the transactions related to that product and a sum of all the transactions transformed into EUR.

For example, for the sample data, the total sum for the product T2006 should be 14,99 EUR.

Requirements

- Do not block the UI while downloading the data from the network. Do not download the data from the network before showing the UI.
- **The use of design patterns, DI, well structured code and project architecture will be valued (do it as you would with a project with a larger scope).**
- Remember that some conversion rates are missing, and they should be derived using the information provided.
- We prefer to get the code in a git repository but please do not put references to bemobile on it.

Tech Stack - i

- You must use Swift and any third party libraries, as you would do with any project.
- iOS min version should be 12.0

Extra points (not required)

- We are dealing with money. Try not to use floating point numbers.
- After each conversion, the result should be rounded to two decimal places, using Bankers' Rounding (http://en.wikipedia.org/wiki/Rounding#Round_half_to_even)
- The right way to specify the request format in HTTP is not using a file extension, but using an "Accept" header. Instead of using the extension try to use the "Accept" header with the right MIME type.
- Show us if you know about unit testing.
- UI Tests.