

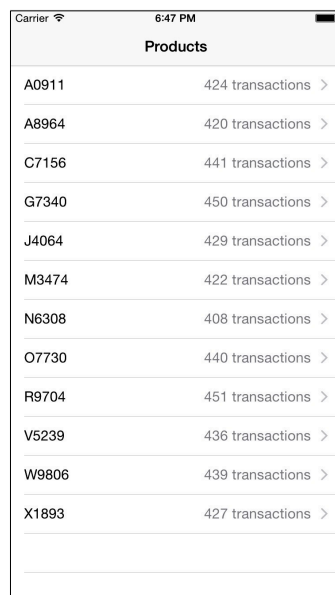
## Transaction viewer

Please make sure you read the full document and think about how to all mission before to do anything:

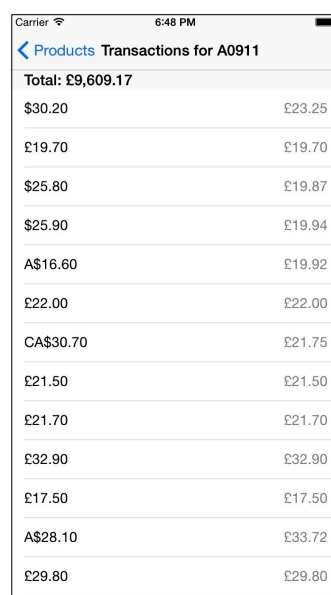
Your manager has asked you to design and implement a mobile application to help the firm executives who are always flying around the globe. They need a list of every product the company trades with, and the sales of those products, which are made in different currencies.

Your application should be composed of two screens:

1. In the first one, your application should
  - give the users the choice of which product they want to see, and
  - show the count of transactions for that product.
2. When the user selects a product, the application must show
  - each of the transactions related to that product,
  - the amount of each transaction converted to GBP and
  - the sum of all of these transactions in GBP.



Products	
A0911	424 transactions >
A8964	420 transactions >
C7156	441 transactions >
G7340	450 transactions >
J4064	429 transactions >
M3474	422 transactions >
N6308	408 transactions >
O7730	440 transactions >
R9704	451 transactions >
V5239	436 transactions >
W9806	439 transactions >
X1893	427 transactions >



Transactions for A0911	
Total: £9,609.17	
\$30.20	£23.25
£19.70	£19.70
\$25.80	£19.87
\$25.90	£19.94
A\$16.60	£19.92
£22.00	£22.00
CA\$30.70	£21.75
£21.50	£21.50
£21.70	£21.70
£32.90	£32.90
£17.50	£17.50
A\$28.10	£33.72
£29.80	£29.80

In a real environment, you will probably download all the data you need from the Internet.

To simplify this task, so you don't need to worry about network management, your application should be designed to be able to work with two property list files:

- **transactions.plist**: holds an array of dictionaries where each one represents a transaction related to a product, indicated by its stock keeping unit (SKU), with a given currency and amount.
- **rates.plist**: specifies the conversion rate from one currency to another.

**Please note that:**

- You should aim to develop a working solution with the simplest possible user interface.
- Some conversions may not be specified in rates.plist. In case they are needed, they should
- be calculated using the known conversions.
- Two different sample data sets are included so you can test your solution with different scenarios.
- Other sets of data will be used to validate your solution, which should be general enough to
- handle all the possible different scenarios, including empty data files and other common errors.

**How are we going to evaluate you?**

- Assume that this application will be delivered to real users. Make sure the app runs and does what we expect from it, be pragmatic.
- The code should be maintainable.
- We will actually read and evaluate your code and decisions **based on the following criteria** :
  - The app needs to work and provide accurate results with various data sets.
  - Algorithm and data structures (chosen algorithm, theoretical background, data structures, alternatives)
  - General architecture (UI, business logic, data model, data import)
  - Unit test
  - Code quality (code readability, following same code style)
  - UI (least important thing to worry about)
- You should be able to solve the problem more or less in the given amount of time.

Good luck!

Two different sample data sets are included so you can test your solution with different scenarios.

- Other sets of data will be used to validate your solution, which should be general enough to handle all the possible different scenarios, including empty data files and other common errors.

### **How are we going to evaluate you?**

- Assume that this application will be delivered to real users. Make sure the app runs and does what we expect from it, be pragmatic.
- The code should be maintainable.
- We will actually read and evaluating your code and decisions **based on the following criteria** :
  - The app needs to work and provide accurate results with various data sets.
  - Algorithm and data structures (chosen algorithm, theoretical background, data structures, alternatives)
  - General architecture (UI, business logic, data model, data import)
  - Unit test
  - Code quality (code readability, following same code style)
  - UI (least important thing to worry about)
- You should be able to solve the problem more or less in the given amount of time.

Good luck!