# Overview of key refactoring carried out on Sprint 1 code.

The code produced during sprint 1 was significantly messy and required heavy refactoring before moving into writing code for the second sprint. The list below indicates key changes made and where the changes occurred (by number) as marked on all relevant code listings.

1. Applied the extract method pattern to the part which calculates the previous Fridays date. Then applied the move method pattern to place the method in a “UtilityFunctions” class intended to group static methods that would lower cohesion if placed in an existing class.
2. Applied the extract method pattern and then the move method pattern to shift the responsibility of creating a query string from the MainActivity to the Stock class. In doing so the Stock class gains the responsibility of defining the historical data it requires.
3. Moved the private DownLoadStockData class required to support threading of downloads from the MainActivity into a new class “DownloadHistoricalData”.
4. Applied the replace array with object pattern creating the “Stock” class.
5. Used the extract method pattern to separate setting up a connection. The for loop to process multiple stocks was left in the “doInBackground” method (calling the method for each iteration).
6. Moved the other section of code for processing a download from the “doInBackgroundMethod” into the “downloadAStock” method.
7. Extracted the part from section 6 which previously added the result information to a string array and modified it to instead set the value within the stock class in line with the change in responsibility for maintaining the data. The substitute algorithm pattern was also used to provide better error handling.
8. Applied the move method pattern moving the convertStreamToString method into “UtilityFunctions” class.
9. Applied extract method pattern then move method pattern to place responsibility for handling view updates to the main activity. Renamed variable name from “tv\_view” to “status”.
10. Applied the replace array with object pattern, merged the resulting object with the “Stock” object as the array values represented parts of a stock. This placed the responsibility of mapping a company name to a stock symbol and tracking the number of units owned by the client into the Stock class.
11. The responsibility of tracking an individual error for a stock symbol was moved into the Stock class. Responsibility for calculating the price of a set of shares in one symbol was moved to the stock class. Responsibility for calculating the total value of the portfolio was moved to the portfolio class. Error checking/handling was improved over the course of the second sprint. The original code in this case was so badly tied together that we made the decision to rewrite those few lines correctly in the designated locations.
12. Used extract method and move method patterns to place converting units from pence to pounds into “UtilityFunctions”.
13. Used extract method and move method patterns to move string formatting with commas into “UtilityFunctions”. Parts which were setting the view elements were extracted and placed into main activity update method.
14. Creating a list of errors encountered with stock data was moved to “Portfolio” and the display of error information moved to the update method in MainActivity. These few lines were substituted with another algorithm that made better use of structural changes.

# Sprint 2 refactoring

Very little refactoring was carried out in producing the code for sprint 2 functionality. The refactoring of sprint 1 code was carried out with an aim of creating a clear and extendable structure, during the course of sprint 2 team members gave strong consideration as to where new functionality belonged within the structure before writing it. As new functionality largely worked with extending existing data structures created in refactoring sprint 1 code the need to move/extract/rename/restructure sprint 2 code was minimal.

A notable exception is the code sections marked 15 in the “DownloadCurrentStockData” and “DownloadHistoricalStockData” classes. Originally the “DownloadHistoricalStockData” class was created from refactoring at points 5 and 6 listed above, a second class for downloading current data was created as the process differs significantly in the request and the result between the different feeds for current and historical data. However the method “downloadAStock” within both classes is the same. This is the main thing that the team is not completely sure about for the moment, the possible refactoring options discussed largely depended on the future development of the project. Ultimately we decided we were ok with this at this point in the development.

# Patterns most used

Most of the refactoring focused on restructuring and separating functionality. To achieve this most refactoring focused on the following patterns:

1. Extract class
2. Extract method pattern
3. Move method pattern
4. Replace array with object pattern
5. Substitute algorithm