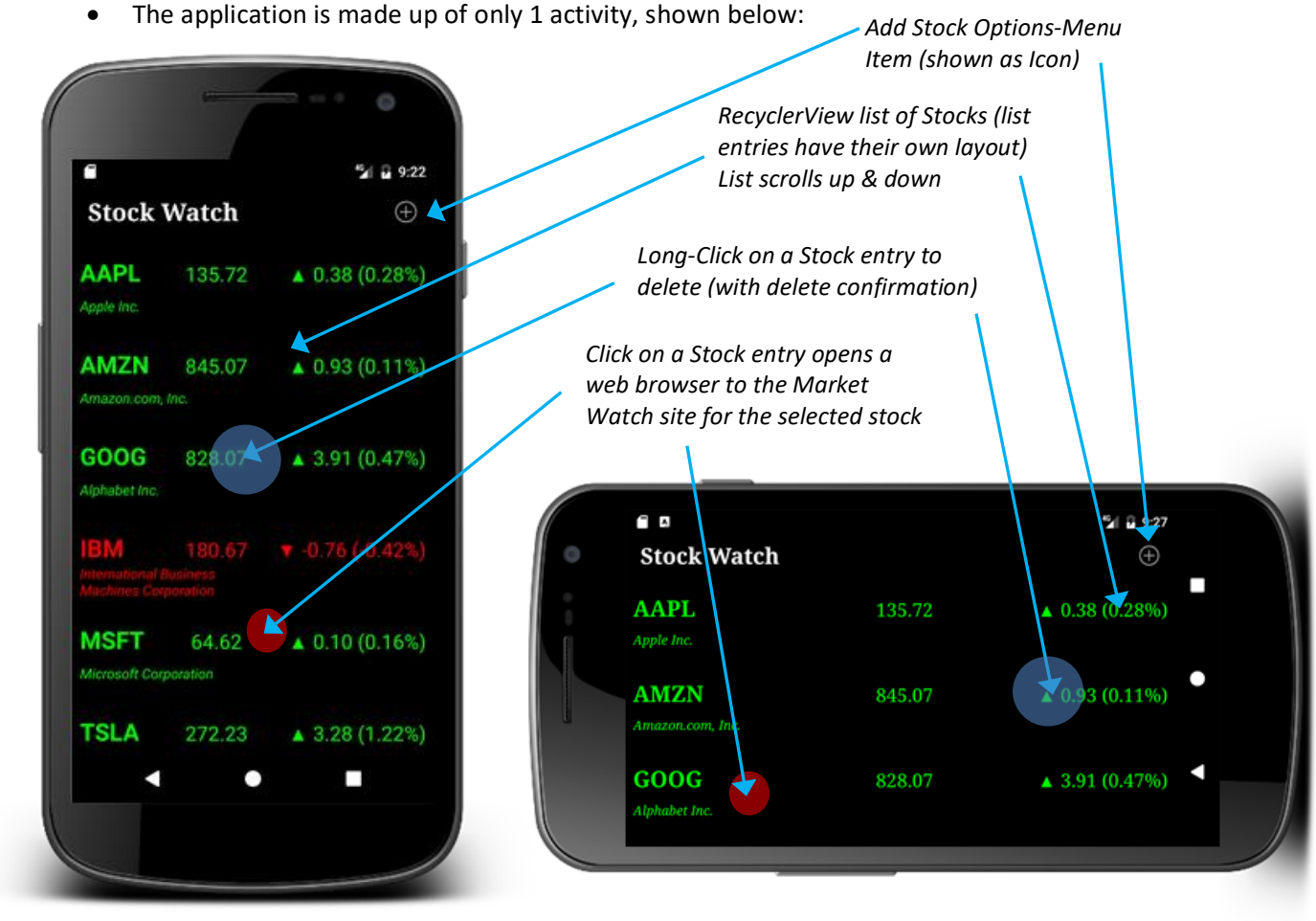
CSC 372/472: Mobile Applications Development for Android

Assignment 3 – Stock Watch *(300 pts)*

Uses: Internet, RecyclerView, Option-Menus, Android Volley, API's, JSON Data, Swipe-Refresh, Dialogs, Implied Intents

App Highlights:

- This app allows the user to display a sorted list of selected stocks. List entries include the stock symbol, company name, the current price, the daily price change amount, and price percent change.
- There is no need to use a different layout for landscape orientation in this application – the same layout should work in any orientation.
- Selected stock symbols and the related names should be stored in a JSON file on the device.
- A Stock class should be created to represent each individual stock in the application. Required data includes: Stock Symbol (String), Company Name (String), Price (double), Price Change (double), and Change Percentage (double).
- Clicking on a stock entry opens a browser displaying the *Market Watch* web page for the selected stock
- Swipe-Refresh (pull-down) refreshes stock data
- The application is made up of only 1 activity, shown below:



A) Internet Data:

Downloading data for a stock symbol requires 2 downloads – one download to acquire the full set of supported stock symbol and company names, and a second download to acquire the financial data for an individual stock.

Download 1: Stock Symbol & Company Data

When started, your app should initiate a download of the full set of supported stock **symbol** and **company names**. This data is saved, and then used whenever the user adds a new stock. *For this you will need an API key*. You can get this by registering at:

https://iexcloud.io/cloud-login?r=https://iexcloud.io/console/home#/register

- Once you provide your account information, you will be presented with various account plans. Scroll to the bottom of the page to the **Get started for free** section and click the "Begin free trial" button.
- Then, go to your email to find the verification and click on the link you find there. Clicking that link will take you to your console.
- Click on the key icon (Access & Security) found along the left side of the page to display the API Token section. There you will find your API token (it begins with "pk_"). You will use that key with your API calls.

Download Source URL: https://cloud.iexapis.com/stable/ref-data/symbols?token=*<Your API key>*

Download Results Example:

Results are returned in JSON format, as a JSONArray of JSONObjects containing the results of the query. The data we are interested in is the stock "symbol" and "name". Below is a small sample of the JSON you will download:


```
[
    ...
  {
    "symbol": "ELA",
    "exchange": "XASE",
    "exchangeSuffix": "",
    "exchangeName": "Nyse Mkt Llc",
    "exchangeSegment": "XASE",
    "exchangeSegmentName": "Nyse Mkt Llc",
    "name": "Envela Corp",
    "date": "2022-10-13",
    "type": "cs",
    "iexId": "IEX_524B375835442D52",
    "region": "US",
    "currency": "USD",
    "isEnabled": true,
    "figi": "BBG000G9WBK0",
    "cik": "0000701719",
    "lei": null
  }, {
    "symbol": "ELAN",
    "exchange": "XNYS",
    "exchangeSuffix": "",
    "exchangeName": "New York Stock Exchange Inc",
    "exchangeSegment": "XNYS",
    "exchangeSegmentName": "New York Stock Exchange Inc",
    "name": "Elanco Animal Health Inc",
    "date": "2022-10-13",
    "type": "cs",
    "iexId": "IEX_513551354C422D52",
    "region": "US",
    "currency": "USD",
    "isEnabled": true,
    "figi": "BBG00LJYS1P8",
    "cik": "0001739104",
    "lei": "549300SHPNDCE059M934"
  }, {
    ...
  }, {

    "symbol": "PAVM",
    "exchange": "XNAS",
    "exchangeSuffix": "",
    "exchangeName": "Nasdaq All Markets",
    "exchangeSegment": "XNCM",
    "exchangeSegmentName": "Nasdaq Capital Market",
    "name": "PAVmed Inc",
    "date": "2022-10-13",
    "type": "cs",
    "iexId": "IEX_4D4D4C3435392D52",
    "region": "US",
    "currency": "USD",
    "isEnabled": true,
    "figi": "BBG00DGWP2L8",
    "cik": "0001624326",
    "lei": "549300V1JXXVXU7P8007"
  },
    ...
]
```

Download 2: Stock Financial Data

When you have the desired stock symbol (and company name), you use the *stock symbol* to download financial data for that stock. *For this you will need the same API key you created in the previous section.*

Query Format: https://cloud.iexapis.com/stable/stock/**STOCK_SYMBOL**/quote?token=**API_KEY**

For example, if the selected stock symbol was TSLA and your API token was pk_123abc, then your full URL would be: **https://cloud.iexapis.com/stable/stock/TSLA/quote?token=pk_123abc**

Download Results:

Results are returned in JSON format, as a JSONObject containing the results data from the query. The data we are interested in is highlighted below (Example using search text "TSLA"):


```
{
  "avgTotalVolume": 72532780,
  "calculationPrice": "close",
  "change": 4.48,
  "changePercent": 0.02062,
  "close": 221.72,
  "closeSource": "official",
  "closeTime": 1665691201054,
  "companyName": "Tesla Inc",
  "currency": "USD",
  "delayedPrice": 221.82,
  "delayedPriceTime": 1665691199299,
  "extendedChange": -1.03,
  "extendedChangePercent": -0.00465,
  "extendedPrice": 220.69,
  "extendedPriceTime": 1665705598902,
  "high": 222.99,
  "highSource": "15 minute delayed price",
  "highTime": 1665691199993,
  "iexAskPrice": 0,
  "iexAskSize": 0,
  "iexBidPrice": 0,
  "iexBidSize": 0,
  "iexClose": 221.83,
  "iexCloseTime": 1665691199661,
  "iexLastUpdated": 1665691199661,
  "iexMarketPercent": 0.020978018385811273,
  "iexOpen": 208.32,
  "iexOpenTime": 1665667800097,
  "iexRealtimePrice": 221.83,
  "iexRealtimeSize": 110,
  "iexVolume": 1919133,
  "lastTradeTime": 1665691199939,
  "latestPrice": 221.72,
  "latestSource": "Close",
  "latestTime": "October 13, 2022",
  "latestUpdate": 1665691201054,
  "latestVolume": 91483045,
  "low": 206.22,
  "lowSource": "15 minute delayed price",
  "lowTime": 1665667981456,
  "marketCap": 694752978377,
  "oddLotDelayedPrice": 221.83,
  "oddLotDelayedPriceTime": 1665691187031,
  "open": 208.49,
  "openTime": 1665667801090,
  "openSource": "official",
  "peRatio": 80.04,
  "previousClose": 217.24,
  "previousVolume": 66860699,
  "primaryExchange": "NASDAQ",
  "symbol": "TSLA",
  "volume": 91483045,
  "week52High": 414.5,
  "week52Low": 206.22,
  "ytdChange": -0.3626764287742008,
  "isUSMarketOpen": false
}
```

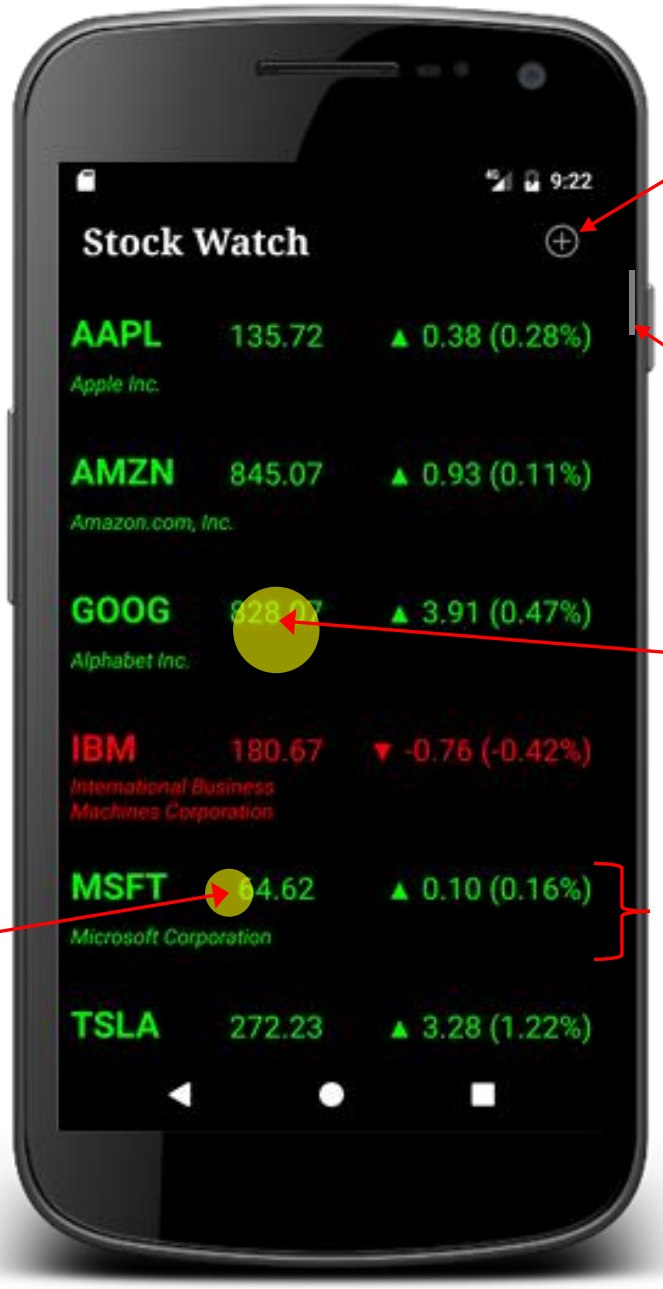

Note – it is possible that the values for "latestPrice", "change", and "changePercent" could be null. If you encounter these null values in your data download, use the value "0.0" for the purposes of our app.

Application Behavior Diagrams

EXTRA CREDIT: All references to the behavior of tapping a stock in the list to open the MarketWatch.com website for the stock are optional and can be done for extra credit (40 points). Instructions related to this extra credit are printed in red.

Each stock entry contains the Stock Symbol (AAPL), the company name (Apple Inc.), the Last Trade Price (135.72), the price change direction (▲ for positive Price Change Amount, ▼ for negative Price Change Amount), the Price Change Amount (0.38), and the Price Change Percentage (0.28%) in parentheses.

If the stock's Price Change Amount is a positive value, then entire entry should use a green font. If the Price Change Amount is a negative value, then entire entry should use a red font.



"Add Stock" action is an Options-Menu with a single item (shown as Icon)

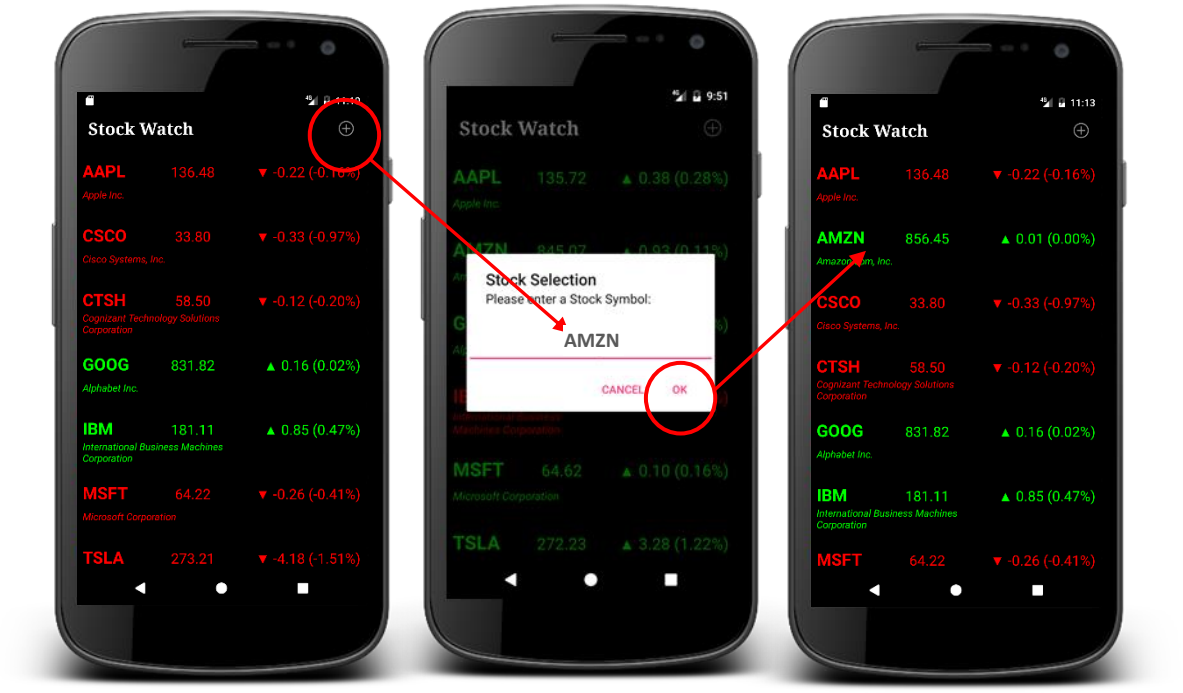
A scrollbar should be present along the right-hand side

Long-Click on a Stock entry to delete (with delete confirmation)

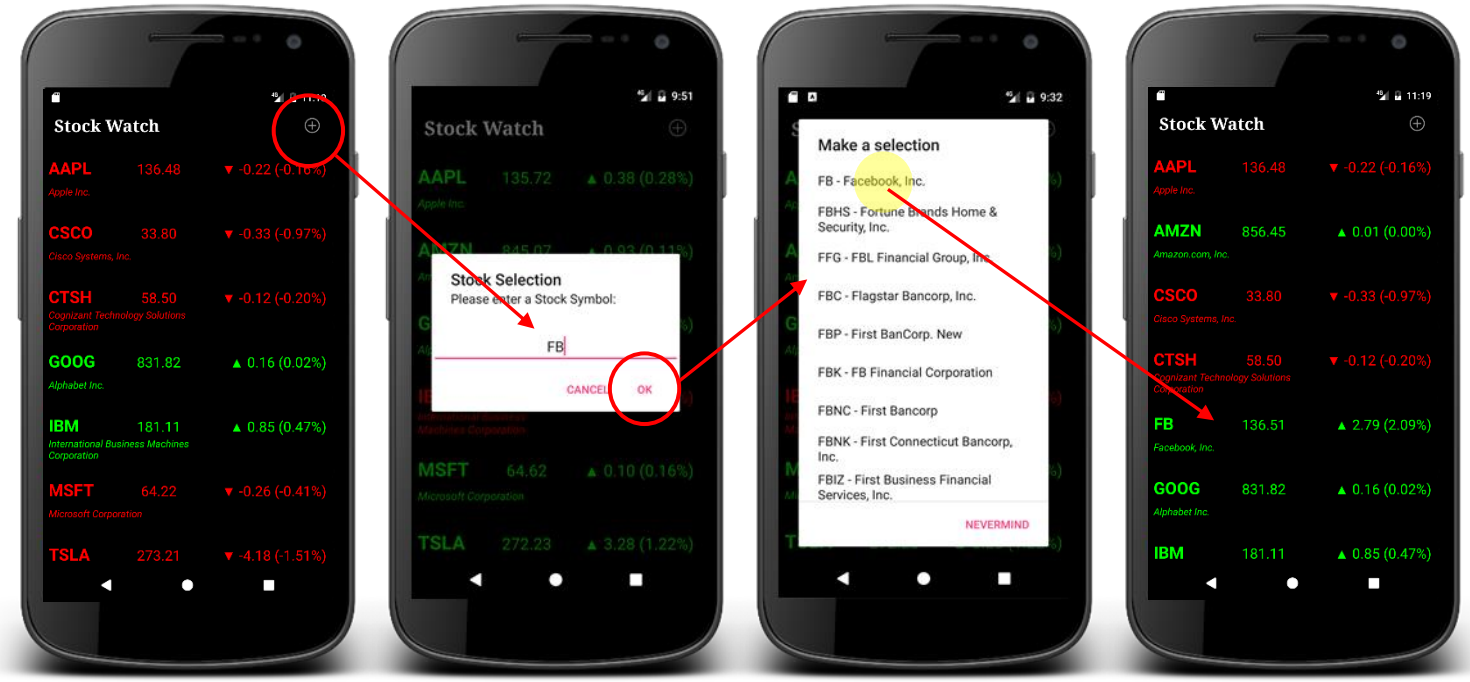
RecyclerView list entries have their own layout

Clicking on a Stock entry opens a web browser to the Market Watch site for the selected stock

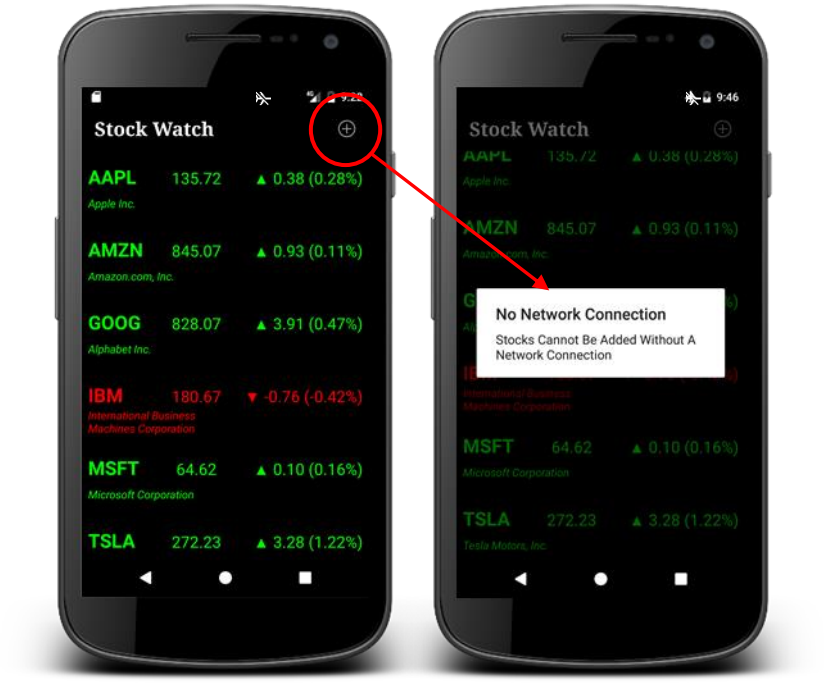
1) Adding a stock – when only ***one* stock matched** the search symbol/name search string *(NOTE: The Stock Selection dialog should only allow capital letters):*



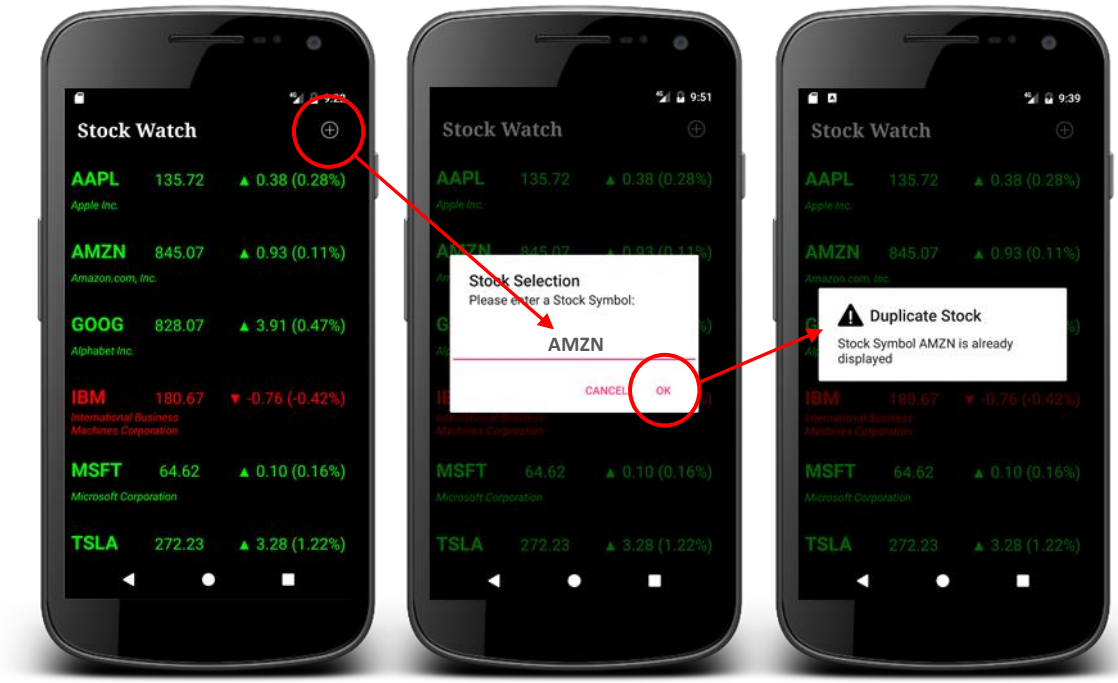
2) Adding a stock – **multiple stocks matched** the search string *(Stock Selection dialog should only allow capital letters, stock selection dialog should display the stock symbol and company name):*



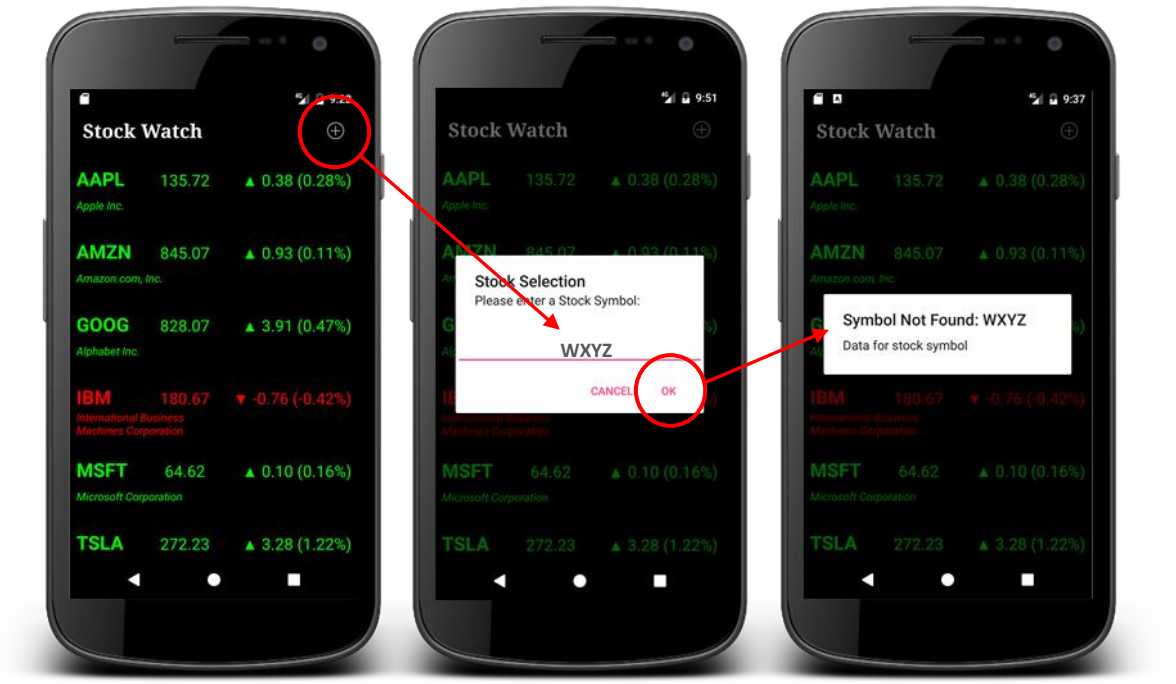
3) Adding a stock with no Network Connection – test using "Airplane Mode" *(You can show no buttons on the error dialog, or an "Ok" button – either is fine):*



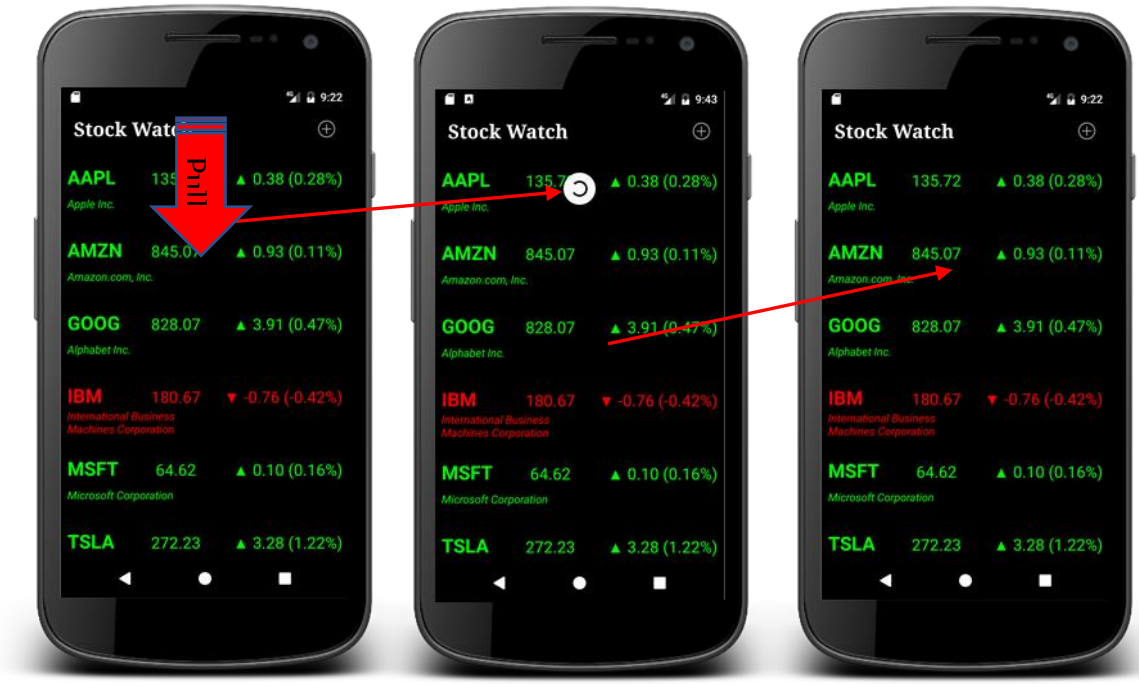
4) Adding a stock – specified stock is a duplicate *(Stock Selection dialog should only allow capital letters, You can show no buttons on the error dialog, or an "Ok" button – either is fine):*



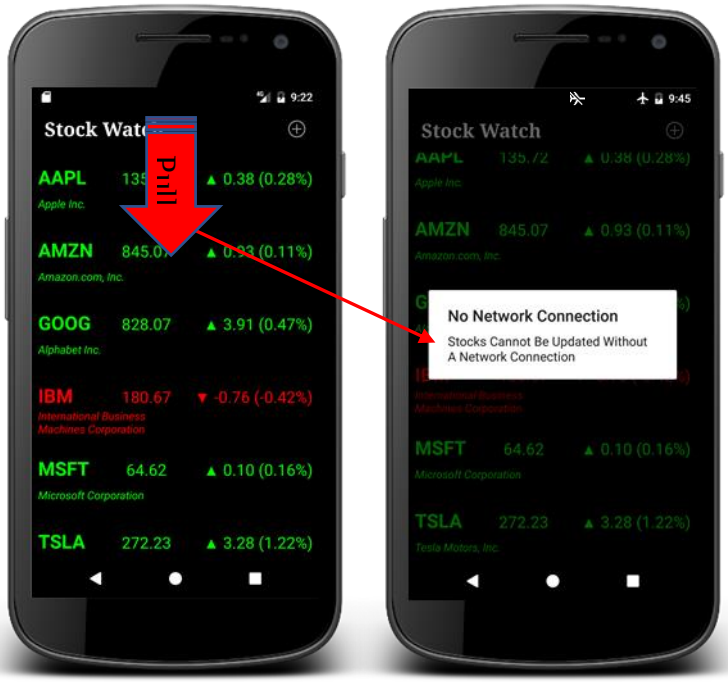
5) Adding a stock – specified stock is not found *(Stock Selection dialog should only allow capital letters, You can show no buttons on the error dialog, or an "Ok" button – either is fine):*



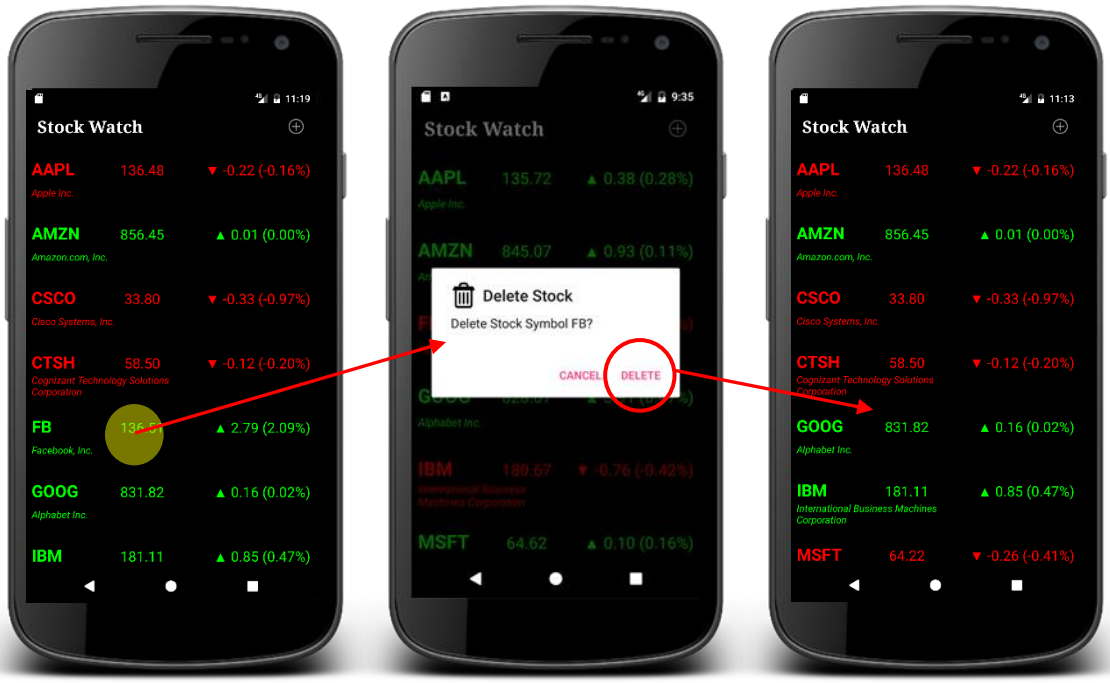
6) Swipe-Refresh (pull-down) reloads (re-downloads) all stock financial data:



7) Swipe-Refresh attempt with no network connection *(You can show no buttons on the error dialog, or an "Ok" button – either is fine):*



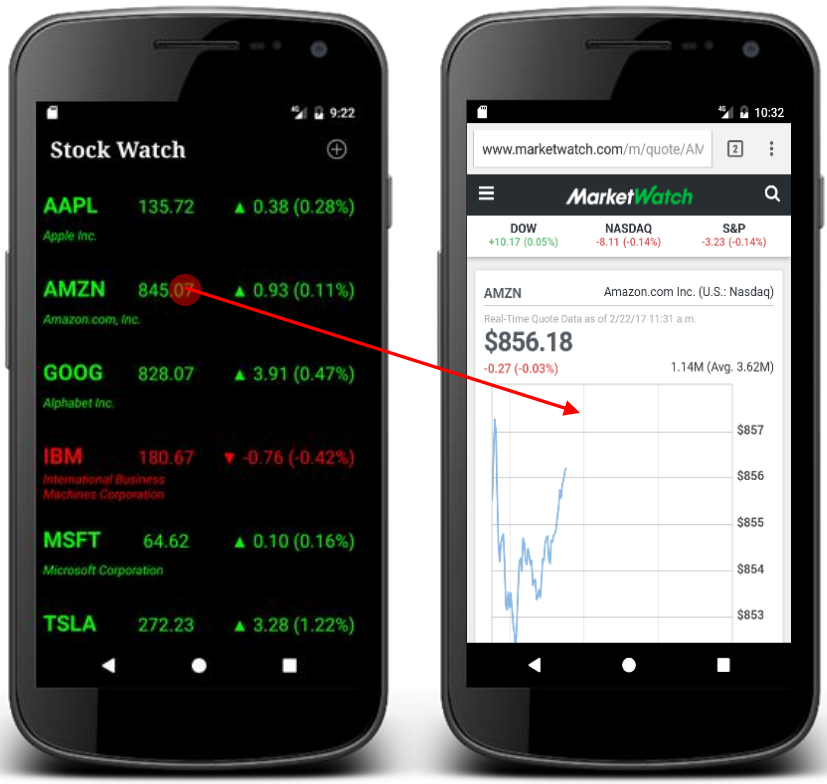
8) Long-Press on a stock to delete it:



9) Extra Credit (40 pts): Tap on a stock to open the *MarketWatch.com* website entry for the selected stock:

MarketWatch URL's are in the form: http://www.marketwatch.com/investing/stock/***some_stock***

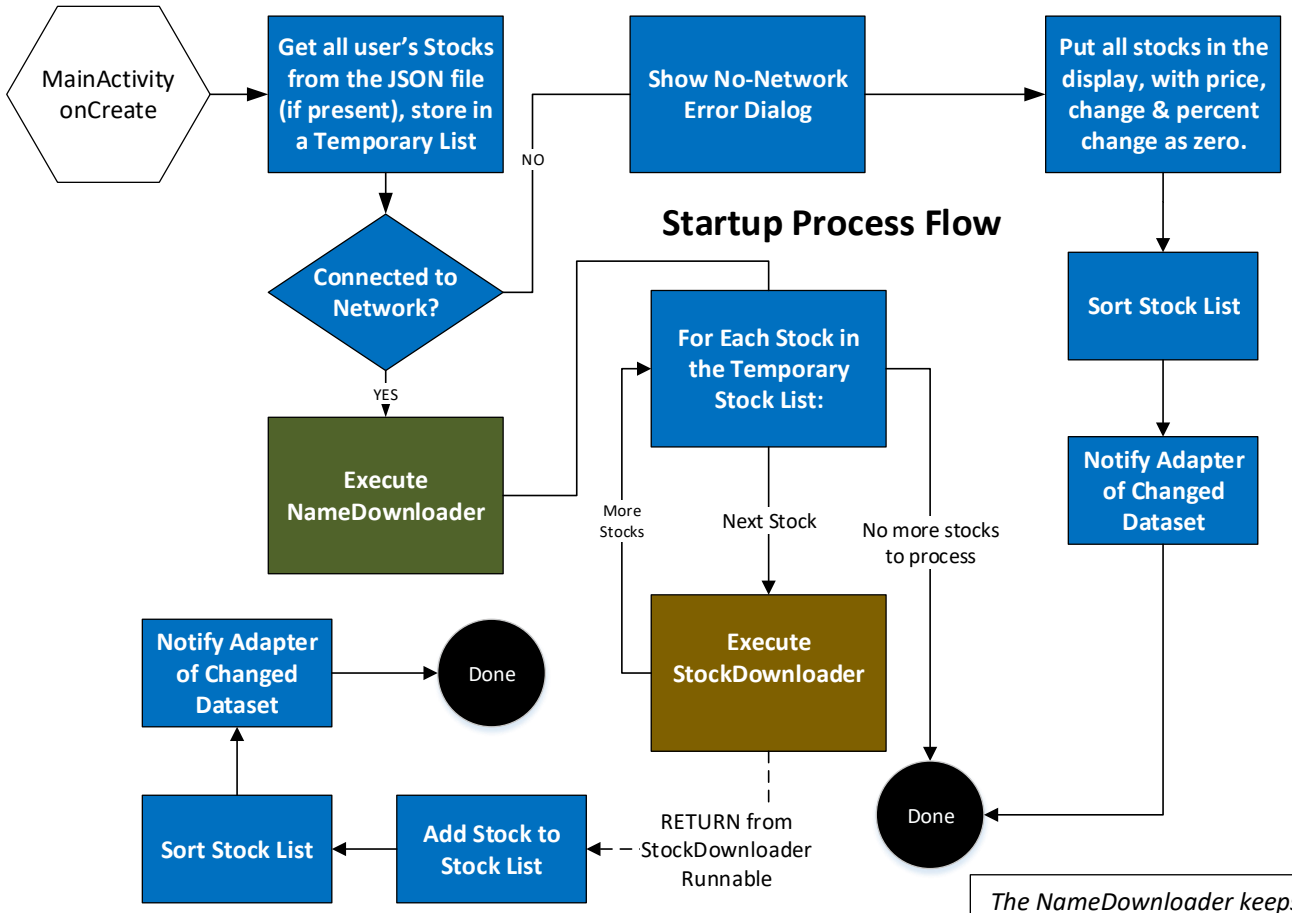
For example: http://www.marketwatch.com/investing/stock/**TSLA**



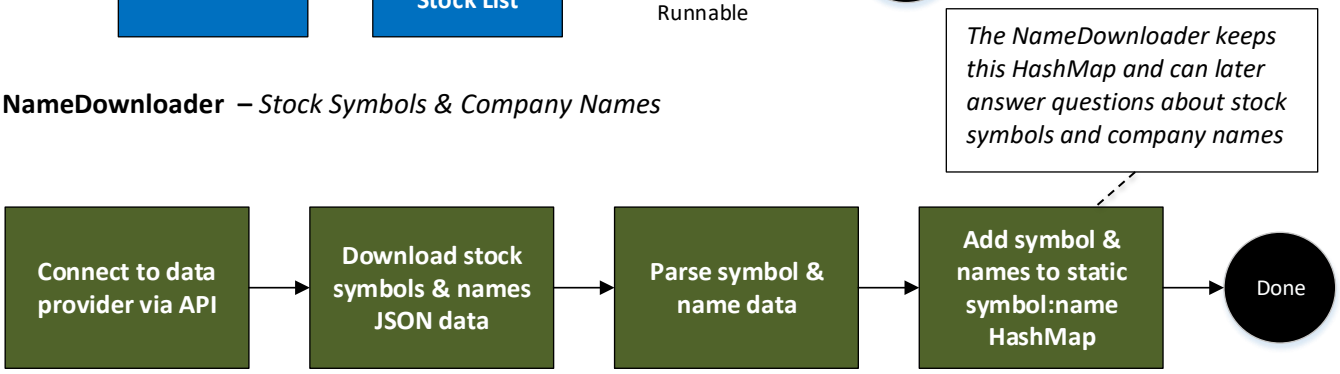
EXTRA CREDIT: All references to the behavior of tapping a stock in the list to open the MarketWatch.com website for the stock are optional and can be done for extra credit (40 points). Instructions related to this extra credit are printed in red.

B) Application Behavior Flowcharts:

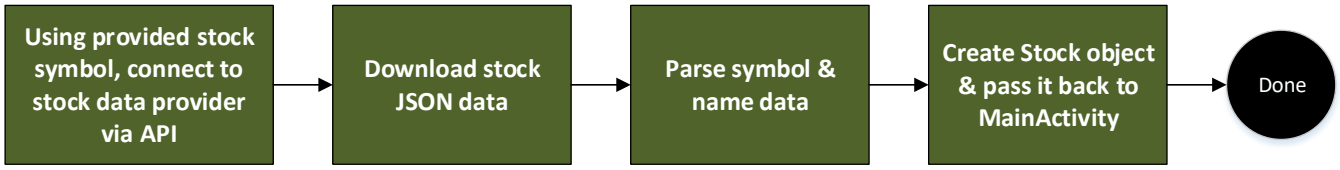
a) App Startup



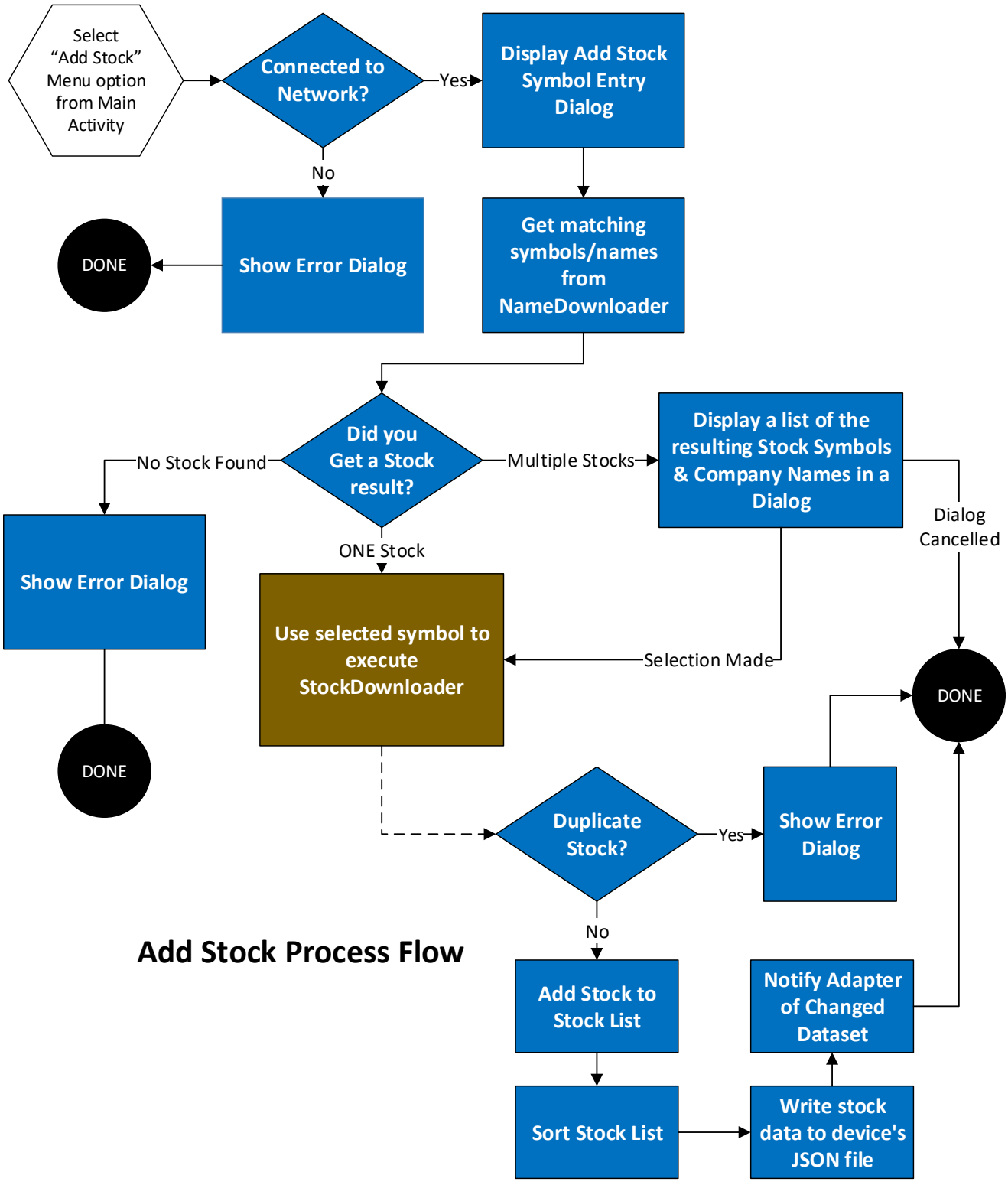
b) NameDownloader – *Stock Symbols & Company Names*



c) StockDownloader – *Financial data for one stock*

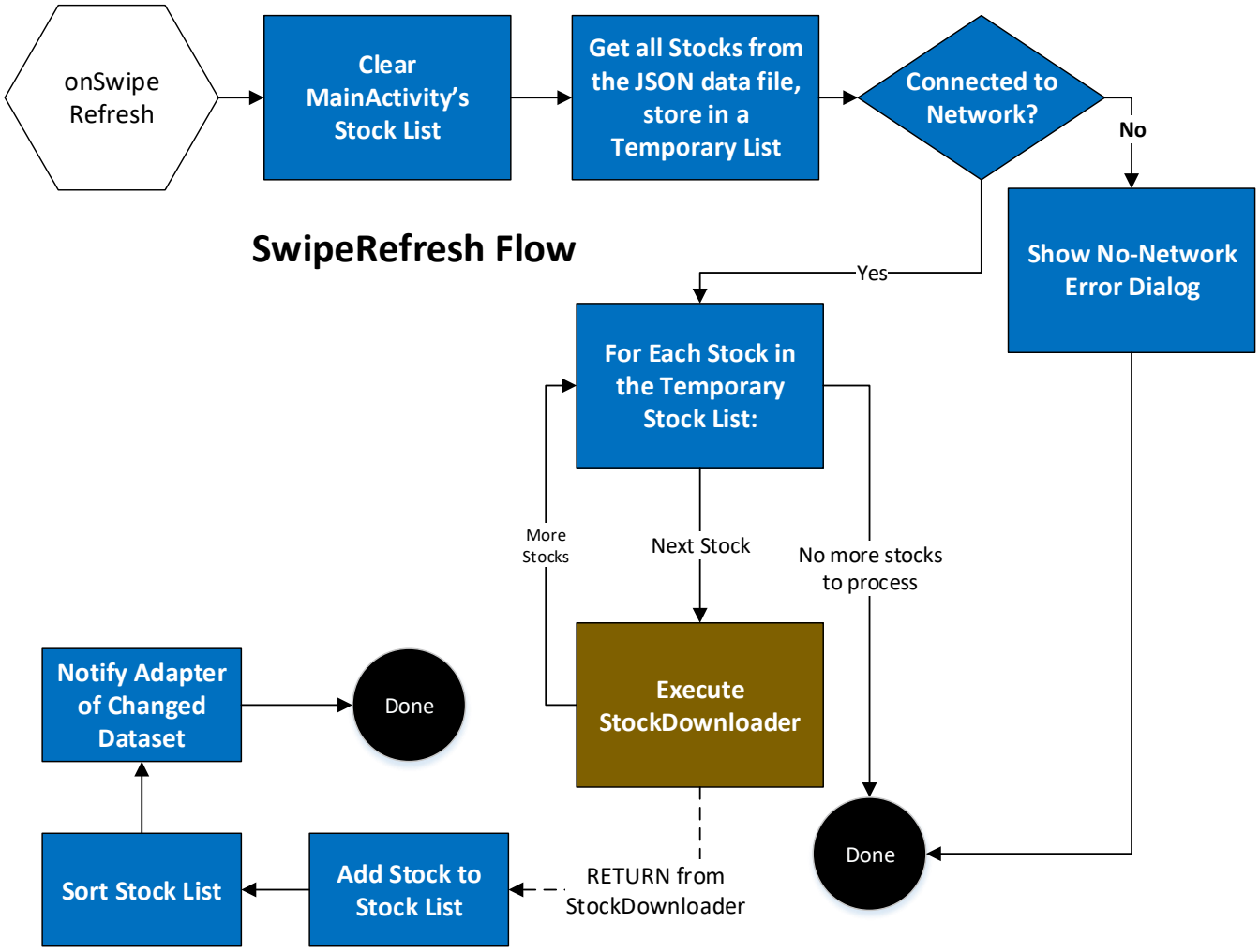


d) Add New Stock

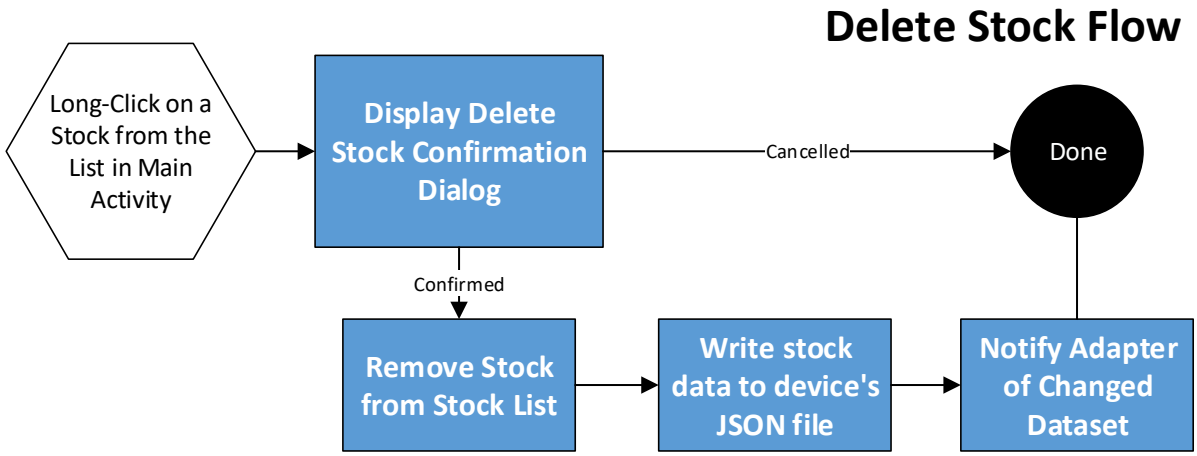


Add Stock Process Flow

e) Swipe-Refresh (pull-down) List:



f) Long-Press Delete Stock:



C) Potential Development Plan

1) Create the base app:
    a. MainActivity with RecyclerView & SwipeRefreshLayout
    b. Create Stock Class
    c. Create RecyclerView Adapter
    d. Create RecyclerView ViewHolder
    e. Create fake "dummy" stocks to populate the list in the MainActivity onCreate.
    f. Add the onClick method. The onClick can open a Toast message for now.
    g. Add the onLongClick methods. The onLongClick should delete the selected entry (using delete dialog).
    h. Add the "add" options-menu that opens dialog and accepts user input. On confirmation you can open a Toast message for now.
    i. SwipeRefresh callback method can open a Toast message for now.

2) Add the data storage elements:
    a. Create code to use the list of stocks to write user's stocks and company names to a JSON file. This is used in the MainActivity when the user adds a stock.
    b. Create code to read the stocks and company names from the JSON file and use those to fill the list of stocks. This is used in the MainActivity's onCreate method.
    c. Code the MainActivity's onClick method to open the browser to the stock's Market Watch site.

3) Add the internet elements and final integration:
    a. Create the Stock Symbol/Company Name downloader/parser. This class should maintain a static HashMap of symbols and company names. Create a public method that accepts a String parameter and returns a list of all symbols and names that match that string parameter. Store these in a HashMap that can be queried later.
    b. Create the Stock Financial Data downloader/parser. Add a method to MainActivity that allows the Stock Financial Data downloader/parser to send the financial data for the selected Stock back to MainActivity.
    c. Implement the Add Stock feature (this uses the results of the above tasks)
    d. Implement the SwipeRefresh callback to re-download the Stock Financial Data for the loaded stocks
    e. Add alerts when startup, add, & refresh are attempted when no internet connection is available.

Assignment Assistance

If you are stuck on an assignment problem that you have exhaustively researched and/or debugged yourself, you can email me a ZIP file of your entire project so that I can examine the problem. All emailed assistance requests must include a detailed description of the problem, and the details of what steps you have already taken in trying to determine the source of the problem.

Note: To make your submission zip file smaller, before zipping your project file, you can remove the ".gradle" folder (found in your project's root directory), and remove the "build" folder (found in the "app" folder in your project's root directory).

Submissions & Grading

- Submissions must consist of your zipped project folder (please execute Build =>Clean Project before generating the zip file).
- Submissions should reflect the concepts and practices we cover in class, and the requirements specified in this document.
- Late submissions will be penalized by 10% per week late. (i.e., from one second late to 1 week late: 10% penalty). No submissions allowed after 1 week late.
- Grading will be based upon the presence and proper functionality of *all features and behaviors* described in this document.
- Grading will be performed with the following SDK details:
  - Project Minimum SDK Version: 25
- Grading will be performed on emulator devices with the following characteristics:

Resolution	Details	Example Emulators to Use
1080 x 1920	With Playstore	Pixel, Pixel 2, Nexus 5, Nexus 5X
1080 x 2220 or 2280	With Playstore	Pixel 3a, Pixel 4

NOTE

This assignment is worth 300 points. This means (for example) that if you get 89% on this assignment, your recorded score will be:

(89% * 300 points = 267 points)

Note that this also means that the 10% late submission penalty will be 10% * 300 points = 30 points.

If you do not understand anything in this handout, please ask.

Otherwise the assumption is that you understand the content.

Unsure? Ask!