CMSC436: Fall 2013 – Week 7 Lab

Objectives:

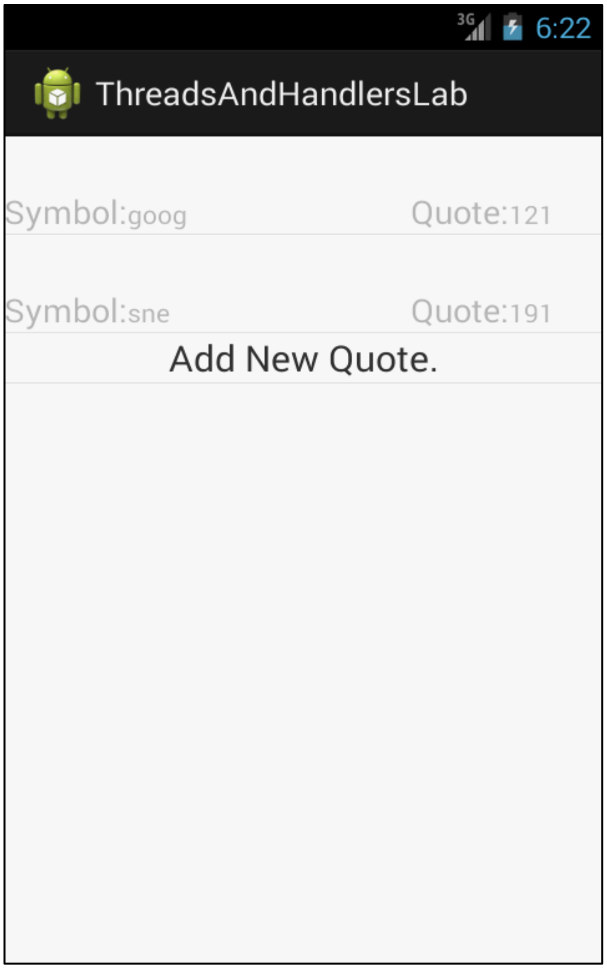
Familiarize yourself with AsyncTasks, and Handlers and prepare the application for networking which will be added in an upcoming Lab. Create and application that uses AsyncTasks and Handlers.

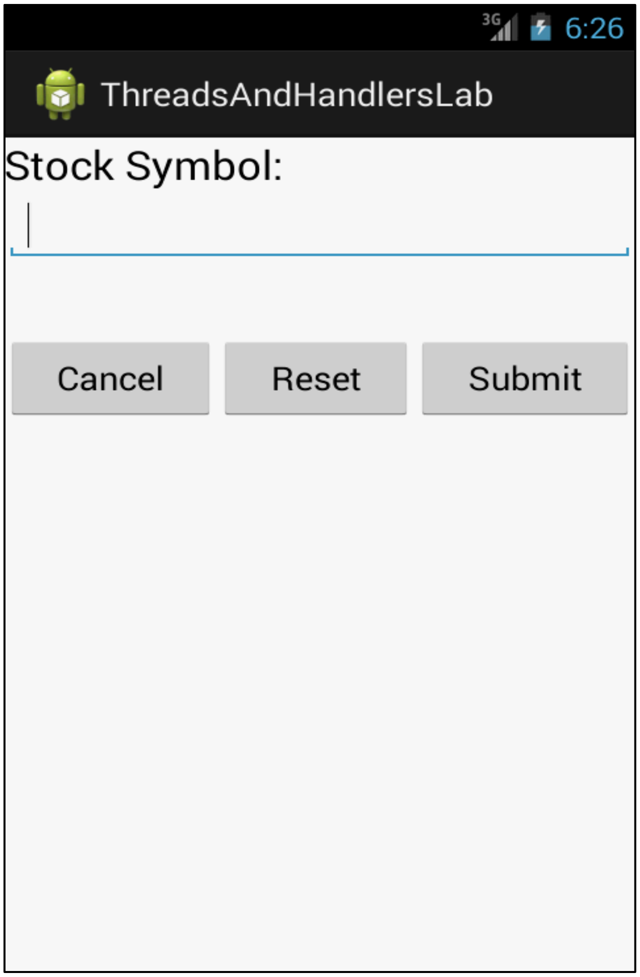
Once you've completed this lab you should have a better understanding of AsyncTasks and Handlers. You should know how these classes work in conjunction with the operations on main thread.

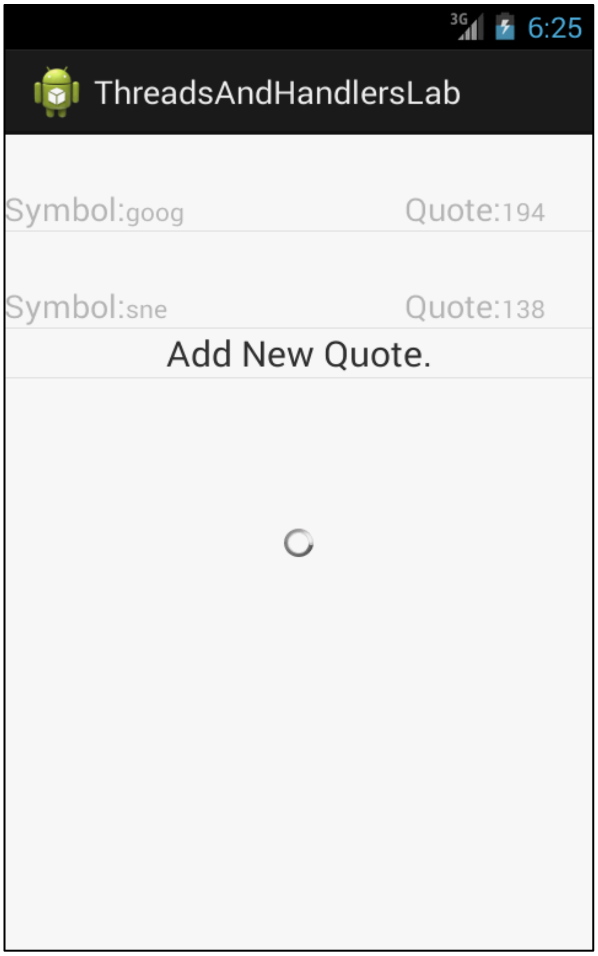
Overview:

This lab has one part. This part uses an AsyncTask to update the Stock quote values and user interface, and uses a Handler to manage the timing of refresh operations.

The application will use the same general user interface that you saw in the BroadcastReceiver lab. The only new thing will be a spinner icon that is provided by a progress bar, which will become visible only while the stock quotes are updating.







Symbol:goog Quote:194

Symbol:sne Quote:138

In the BroadcastReceiver lab, we used an ExecutorService to update stock quotes every 15 seconds. In this part, we will get rid of the ExecutorService and replace it with an AsyncTask and Handler. In other words, you will use an AsyncTask and Handler to update the stock prices and the UI.

Implementation Notes:

1. Download the application skeleton files from the Lectures & Labs web page and import them into your IDE. The skeleton provides most of the app's implementation except for the AsyncTask and Handler.

2. In StockQuoteActivity.java, go to TODO1 item and implement a Runnable called mHandlerTask, which will do the following:

   - Create a new instance of the AsyncTask,
   - Execute the AsyncTask, and
   - If mIsRefreshing and mIsForeground are true, reschedule this Runnable so that it restarts in 15 seconds.

3. In StockQuoteActivity.java, go to TODO2 item and implement an AsyncTask, which will do the following:

   - onPreExecute() - update the UI to make the progress bar visible,
   - doInBackground() - Update the prices of each stock and return the updated prices,

- onPostExecute() – Receive the updated prices and update the user interface with these new prices. Update the UI to hide the progress bar.

Deliverables: Your source code project