CS180 Lab 12: Android 3: Simple Concurrent Program

Sections from the textbook relevant for this lab: 13.4, 13.5, 13.6, 13.7,14.4.1,14.4.2
Lab for week: 12
Lab created by: John Franklin Jr.

Learning Objectives

1. Create a simple concurrent program.

Objectives

- Setup files and directories.
- Query "server" to get stock price.
- Update local price to show on screen.

Files

`StockUpdater.java`

Setup

Make a new directory for Lab 12:

```
$ cd
$ cd cs180
$ mkdir lab12
$ cd lab12
$ pal lab12
$ drjava StockUpdater.java &
```

Information

The stock information that is currently shown on the screen is held in an array of `Stock` objects in a `static` array. The array name is `Stock.stock_array`. A `Stock` object has three public variables, `name, price, and price_change_percent`. The `name` variable is of type `String` and the other variables are of type `double`. All stock information will be retrieved by calling

the static function `Stock.getStockPrice` (**`String`** name).This method returns a Stock object with the name of the stock requested, the current price of the stock, and price_change_percent is set to zero.

Exercise 12-1: Check for New Stocks

Objective: Constantly check for new stocks that are added to `Stock.stock_array`.

Steps:

1. Within the loop use the `Thread.sleep` (long) method to sleep for 40ms.
2. Within the `main` method, create an infinite loop that checks if the `Stock.stock_array` length has changed.
3. If the length of `Stock.stock_array` has changed after 40ms find the new `Stock` that is added to the list. (Stocks are added to the end of `Stock.stock_array`.)

This step will not produce anything new on the phone. Complete the next exercise to produce something on the Android phone.

Exercise 12-2: Creating Threads

Objective: Create a new thread that will update the `Stock` objects within the `Stock.stock_array` array.

Steps:

1. Create a new class that extends `Thread`.
2. Create a constructor that takes an integer as a parameter. The integer passed as a parameter should hold the position in the array of the `Stock` that this thread will update.
3. In the `run` method create a loop that will run constantly calling the method `Stock.getStockPrice` (**`String`** name).
4. Use the `Stock` object returned to retrieve the price from the `Stock` object.
5. Get the current price of the stock within the array and find the percent change in the price in the array and the price given by the method `Stock.getStockPrice` (**`String`** name).
6. Update the `Stock` object in the array with the new price and percent change in the price.
7. Call the `StockUpdater.updateList()` method from the StockUpdater class to make the changes made to stock array visible on the screen.
8. Within the loop created in exercise one, when a new stock is added to `Stock.stock_array` start a new thread of the class created in this exercise.

Grading

Criteria	Percent

Exercise 1	40%
Exercise 2	60%

Turn In

Show your TA your application for a grade. There will be nothing to turn in for this lab.

<End of lab 10>