Location Lab

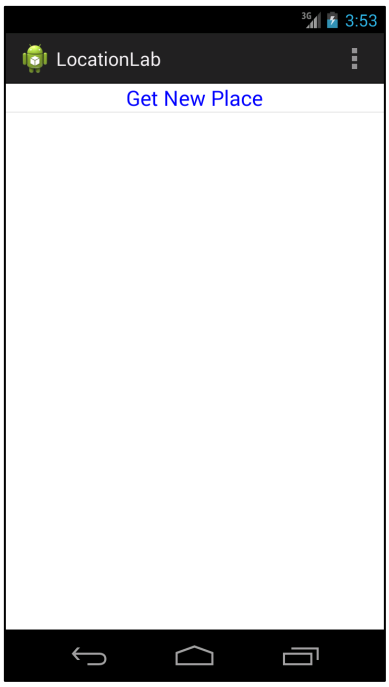
Use Location information within your app.

Objectives:

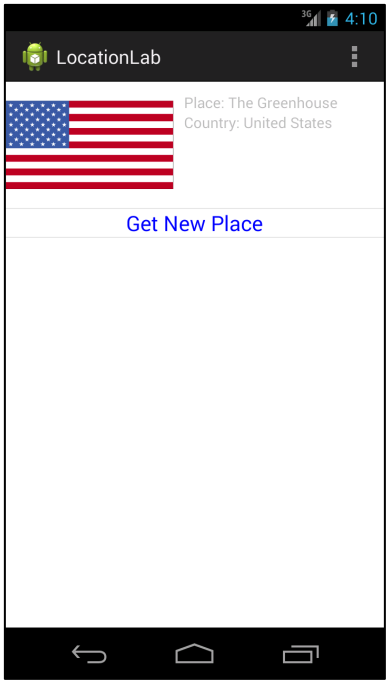
This week's lab will help you learn how to use location information in your Android applications. Upon completion of this lab, you should have a better understanding of how to listen for and respond to Location measurements.

This application displays a ListView containing a set of Place Badges. Each Place Badge contains a country flag, a country name, and a place name corresponding to the user's location when the Place Badge was acquired. The Footer for the ListView displays the words "Get New Place." When the user clicks on the Footer, the application will attempt to capture a new Place Badge based on the user's current location.

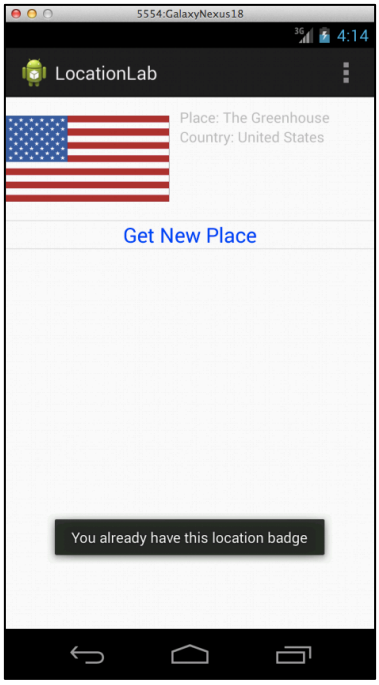
When the application starts the user will not have any Place Badges, so the LIstView will be empty, as shown below:



If the user clicks on the Footer and the application does not already have a Place Badge for a location within 1000 meters of the user's current location, then the application should create and execute an AsyncTask subclass called PlaceDownloaderTask that acquires the data needed to create the Place Badge. Once the Place Badge information has been acquired, the Place Badge should appear in the ListView.



If the user clicks on the Footer, but the application already has a Place Badge for a location within 1000 meters of the user's current location, then the application should display a Toast message with the text, "You already have this location badge."



In addition, the application should gracefully handle or better yet prevent the user clicking on the Footer when the application has not acquired a valid user location.

To implement this application, you will need to acquire location readings from Android. How you implement this is up to you, however, your app will need to listen for location updates from the NETWORK_PROVIDER (which we will control for testing purposes). Be aware that listening for location updates from other providers is likely to cause problems during testing.

Implementation Notes:

1. Download the application skeleton files and import them into your IDE.
2. Complete the TODO items, most of which are in the PlaceViewActivity.java file. There is also one TODO in PlaceDownloaderTask.java.

Testing:

The test cases for this Lab are in the LocationLabTest project, which is located in the LocationLab/SourceFiles/TestCases/LocationLabTest.zip file. You can run the test cases either all at once, by right clicking the project folder and then selecting Run As>Android Junit Test, or one at a time, by right clicking on an individual test case class (e.g., TestOneValidLocation.java) and then continuing as before. The test classes are Robotium test cases. You will eventually have to run each test case, one at a time, capture log output, and submit the output to Coursera. These test cases are

designed to drive your app through a set of steps, passing the test case is not a guarantee that your submission will be accepted. The TestOneValidLocation test case should output exactly 4 Log messages. The TestSameLocation test case should output exactly 6 Log messages. The TestTwoValidLocations test case should output exactly 8 Log messages.

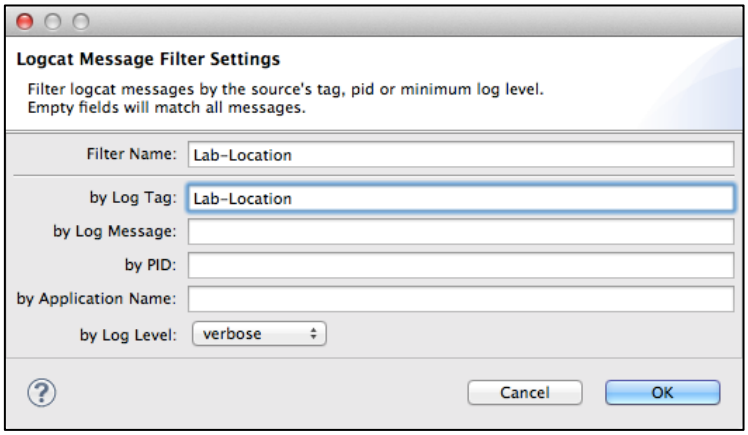
Warnings:

1. These test cases have been tested on a Galaxy Nexus AVD emulator with API level 18. To limit configuration problems, you should test you app against a similar AVD.
2. Our MockLocationProvider relies on a method that was included in API level 17. Therefore, the TestCases will fail to compile on earlier platforms.
3. During testing you should not provide your own location information.
4. The application requires a working network connection. You will also need to create an account at http://www.geonames.org/login. Your username will need to be updated in PlaceDownloaderTask.java.

Once you've passed all the test cases, submit your log information to the Coursera system for grading.

Tips: Saving a LogCat filter.

1. In the LogCat View, press the green "**+**" sign to "add a new LogCat filter."
2. A dialog box will pop up. Type in the information shown in the screenshot below.
3. A saved filter called, "Lab-Location" will appear in the LogCat View.



Tips: Running your test cases and capturing your LogCat output for submission.

1. For each test case, clear the LogCat console by clicking on the "Clear Log" Button (the Button with the red x in the upper right of the LogCat View).
2. Then right click on an individual test case file in the Project Explorer. Run As -> Android JUnit Test.
3. When the test case finishes, if it's not already selected, click on the "Lab-Location" filter in the left hand pane of the LogCat View.
4. Select everything within the LogCat View (Ctrl-A or Cmd-A) and press the "Export Selected Items To Text File" button (floppy disk icon) at the top right of the LogCat View.
5. Submit this file to the Coursera system.

If you get through Exercise A and submitted and passed the tests, and feel that you'd like to do more,

here are some suggested additions. This is optional and ungraded.

Optional Exercise B: More Sophisticated Location Management

Modify your application so that you listen for location updates from multiple location providers. Experiment with different criteria for decided when to use a location reading, when to turn on and turn off location readings, etc.

Optional Exercise C: Using Gestures

Modify your application so that you use your own custom gesture, rather than clicking on the Footer to get a new Place Badge. Better yet, if you have a device or can install a Sensor simulator, such as the one found here: http://code.google.com/p/openintents/wiki/SensorSimulator, use the accelerometer to recognize a shake motion (physically shaking the device) as the signal to get a new Place Badge.