Location Lab

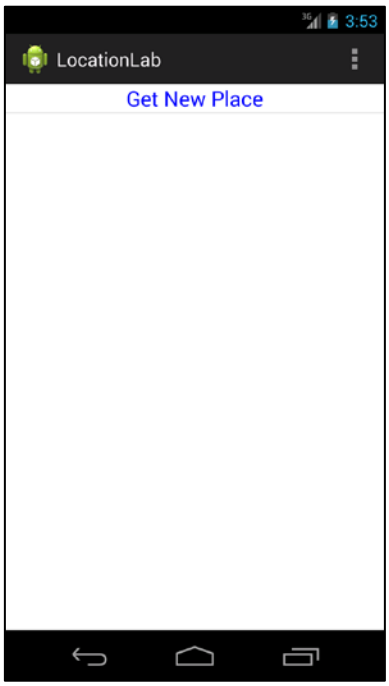
Use Location information to make your app context-aware.

Objectives:

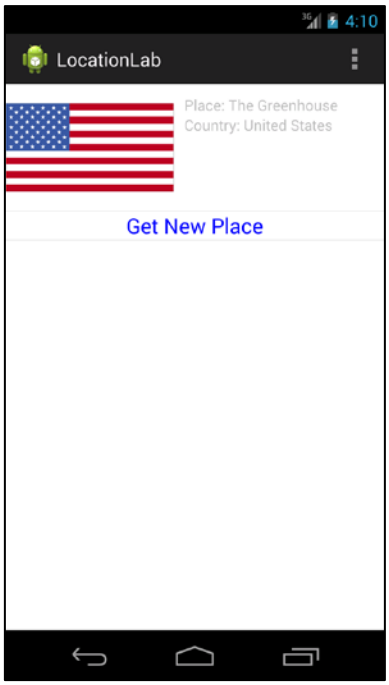
This week's Lab will help you learn how to use location information in your Android applications. Upon completion of this lab, you should have a better understanding of how to listen for and respond to Location measurements.

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:

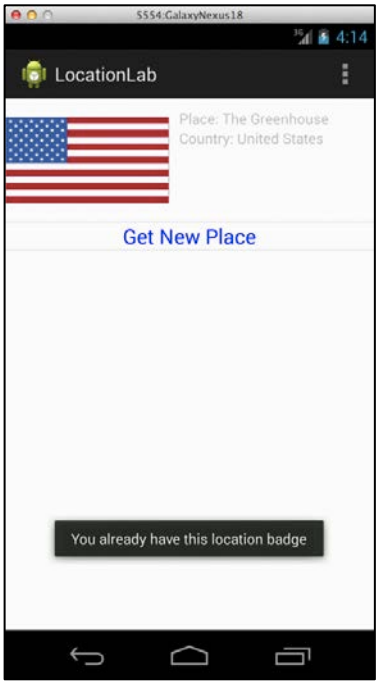


If the user clicks on the Footer and the application does not already have a Place Badge for a location within 1000 meters of the user's current location, then the application should create and execute an AsyncTask subclass called PlaceDownloaderTask that acquires the data needed to create the Place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



If the information does not include a country name, then the application should issue a Toast message with the text, "There is no country at this location".

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.

See the LocationLabScreencast.mp4 screencast to see the app in action.

To implement this application, you will need to acquire location readings from Android. Exactly 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). You can inject locations into the app, by clicking on the menu and selecting one of the menu items, specifically "Place One", "Place Two," or "Place No Country." 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 optional TODO in PlaceDownloaderTask.java.

Testing:

The test cases for this Lab are in the Lab8_LocationLabTest project. 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 and then continuing as before. The test classes are Robotium test cases.

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 first included in API level 17. Therefore, the TestCases will fail to compile on earlier platforms.
3. You will need to go into your device's Developer Options and make sure that you've enabled "Allow Mock Locations."
4. During testing you should not provide your own location information. To inject locations you will need to click on menu options, specifically, "PlaceOne," "PlaceTwo" and "PlaceNoCountry."
5. The application screenshots show above require 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. If you do not have a working network connection, you can still use the app, but the flag images will use a simple placeholder.

Submission

To submit your work you will need to submit the LocationLab project files we've asked you to modify. These files should be stored in specific directories as described below and then compressed in a zip file. Then you will submit this zip file to the Coursera system. The automatic grading system will test your submission and give you feedback. This process may take some time, especially if many students are submitting at the same time.

To make sure your submission is correctly graded, pay attention to the following aspects:

1. Your project files must be in a zip file named LocationLabSubmit.zip.
2. The zip file should have the following directory structure:

LocationLabSubmit/
- LocationLab/
    - PlaceViewActivity.java
    - PlaceDownloaderTask.java

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.