ChihYung Wu

4. 12.2016

Assignment 4

The deliverables are:

# Part 1—Overall Project Description:

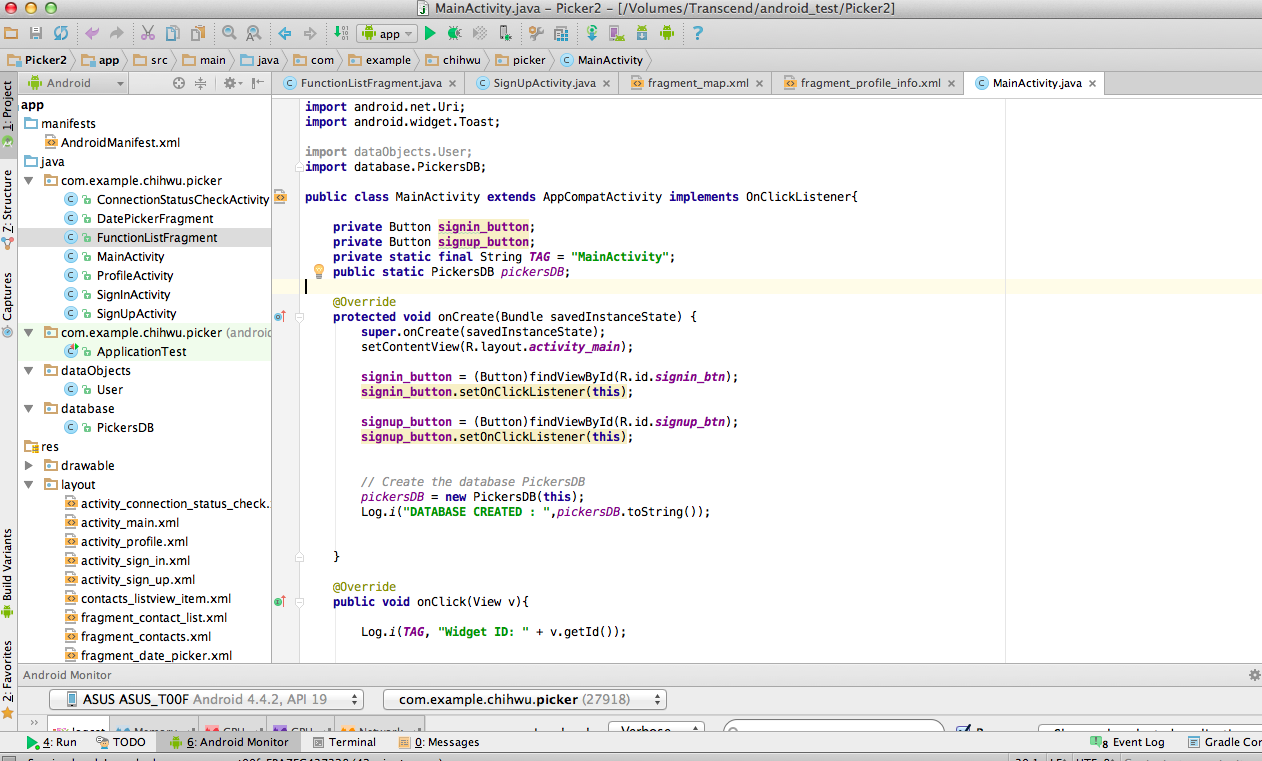
Provide a paragraph (no more) outlining the Android application you’d like to build for this course. Don’t worry about being overly ambitious or not completely specific to begin with: we hold you only to the specific requirements in Part 3.

# This will be a cool-friend-making application that will allow its users to quickly find and make friends with someone in their proximity. In order to be able to use this application, the user will have to create an account by inputting their username, password, date-of-birth, introduction. The results will display the info of all the users in that user’s proximity.

# The application can allow users to sign up and sign in. After they sign in, users can see their profile info, and the friends on their contact list for future location detection purpose.

# Currently, users can use Google map to simple mapping function in a web browser. But later on, Google Map api will be used to embed the google map service.

# Part 2—Screenshot of the directory structure (project or package):

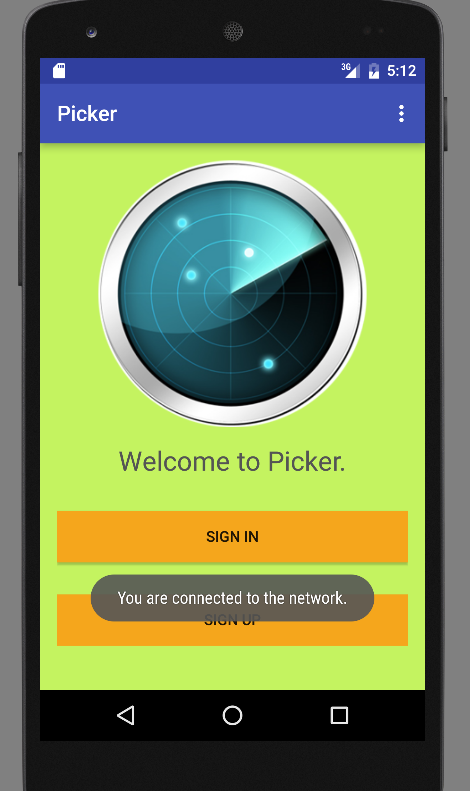
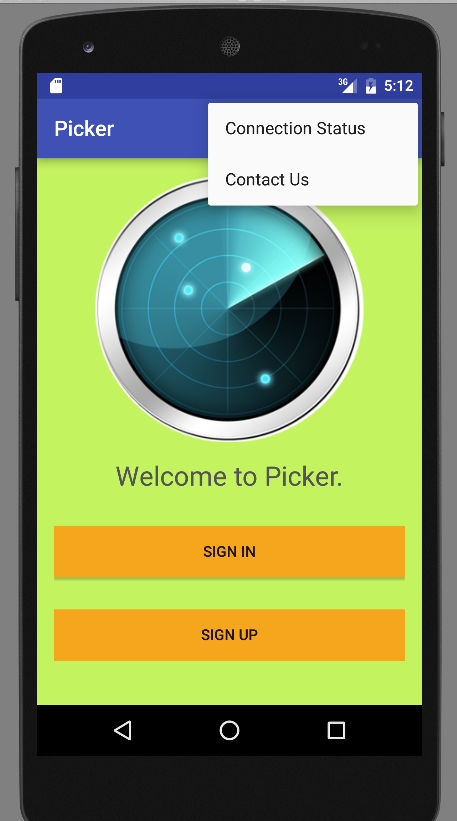


# Part 3—Application Features with output and tests: *(maximum 2 pages of 12-point text, including figures)*

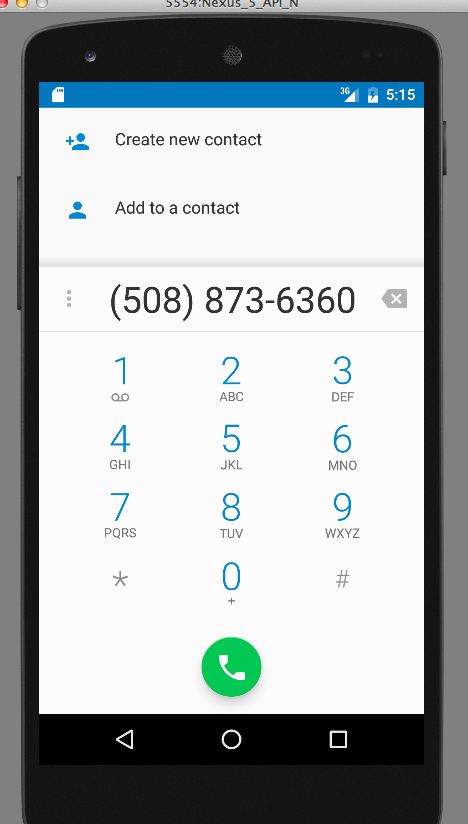
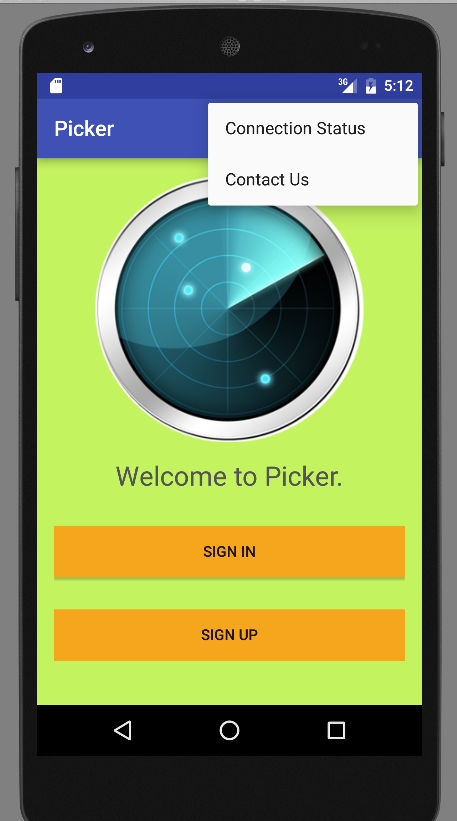
Specify the features of your application that you implemented. The priority is to get practice with as many as possible of the Android constructs covered in Module 1. Number your features 1, 2, 3, … . Each feature must accompanied by output or screen shot that show you accomplished it.

(Features based on elements covered in this week’s module)

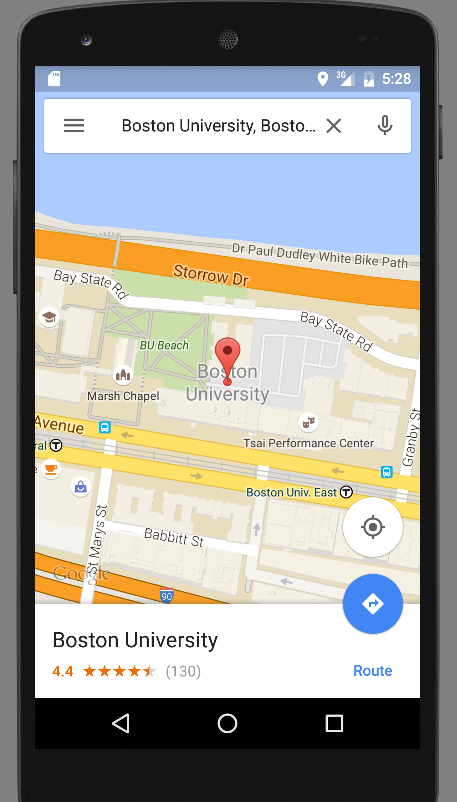
Application Feature 1: (Users can use System Service to check their network status)



Application Feature 2: (Users can use Intent Action to dial phone our company’s phone number)



Application Feature 3: (Users can use Google Map to view its current location.)

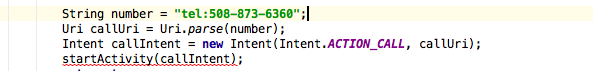


# Part 4—List of Android Elements: *(maximum 1 page of 12-point text)*

Use as many of the Android elements (e.g., built-in keywords, data types or functions) as possible covered in module 1. Provide lines from your code and where they were used once.

(Elements covered in this week’s module)

**New Element #1 (Use of Intent for StartActivity() method )**



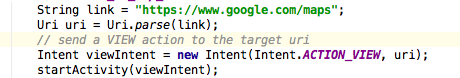
**New Element #2 (Use of Intent for StartActivityForResult() method )**

Macintosh HD:Users:ChihWu:Desktop:Screen Shot 2016-04-12 at 4.58.23 AM.png

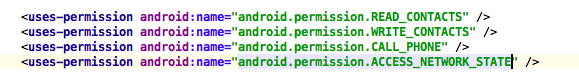
**New Element #3 (Use of explicit Intent )**

**Macintosh HD:Users:ChihWu:Desktop:Screen Shot 2016-04-12 at 5.03.41 AM.png**

**New Element #4 (Use of implicit Intent )**



**New Element #5 (Use of Permission in AndroidManifest.xml )**

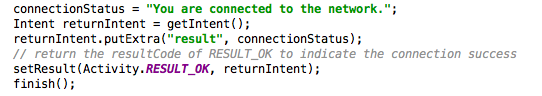


**New Element #6 (Use of Intent filter/action element )**

Macintosh HD:Users:ChihWu:Desktop:Screen Shot 2016-04-12 at 5.08.33 AM.png

Macintosh HD:Users:ChihWu:Desktop:Screen Shot 2016-04-12 at 5.08.51 AM.png

**New Element #7 (Use of Extras for Intent)**



**New Element #8 (Use of System Service )**

Macintosh HD:Users:ChihWu:Desktop:Screen Shot 2016-04-12 at 5.10.43 AM.png

# Part 5—Example Code

Provide a page—in 10-point Ariel Narrow font—of your best (clearly documented)[[1]](#footnote-1) code. You may precede it with a paragraph (not counted in the page limit) explaining where and how it fits with the rest of your project.

**FunctionListFragment.java has all the logics for the tab functions of my app and all the functions my app provides so far.**

FunctionListFragment.java

**package** com.example.chihwu.picker;  
  
  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.database.Cursor;  
**import** android.net.Uri;  
**import** android.os.Bundle;  
**import** android.support.v4.app.Fragment;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.Button;  
**import** android.widget.ListView;  
**import** android.widget.SimpleAdapter;  
**import** android.widget.TabHost;  
**import** android.widget.TextView;  
**import** android.util.Log;  
**import** java.util.ArrayList;  
**import** java.util.HashMap;  
**import** android.view.View.OnClickListener;  
  
**import** dataObjects.User;  
**import** database.PickersDB;  
  
  
**import** android.provider.ContactsContract.Contacts;  
**import** android.provider.ContactsContract.Data;  
**import** android.provider.ContactsContract.CommonDataKinds.Phone;  
  
  
*//Known issues: Please note that for the content provider for the contacts to work properly,  
// one needs to use a real android device rather than the emulator.  
// In addition, the function for the Search tab is still not implemented yet as it requires the use  
// GoogleMap API.***public class** FunctionListFragment **extends** Fragment **implements** OnClickListener {  
  
 **private** TextView **functionTextView**;  
 **private** String **currentTabTag**;  
  
 */\*Variables for Profile Info Tab\*/* **private** TextView **user\_name\_txtView**;  
 **private** TextView **user\_firstName\_txtView**;  
 **private** TextView **user\_lastName\_txtView**;  
 **private** TextView **user\_password\_txtView**;  
 **private** TextView **user\_email\_txtView**;  
 **private** TextView **user\_dob\_txtView**;  
 **private** TextView **user\_intro\_txtView**;  
 */\*\*\*\*\*/  
  
 /\*Variables for the contacts Tab\*/* **private final** Uri **DATA\_URI** = Data.***CONTENT\_URI***; *//the CONTENT\_URI is a constant containing the Contacts ContentProvider's info from the class android.provider.ContactsContract* **private** ListView **contact\_list\_view**; *// for displaying all the contacts in a ListView in the tab Contacts of this fragment  
 /\*\*\*\*/  
  
  
 /\*Variables for the search Tab\*/* **private** Button **show\_on\_map\_btn**; *// for displaying Google map in a Web browser when clicked  
  
 /\*\*\*\*/* @Override  
 **public** View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {  
 TabHost tabHost = (TabHost) container.getParent().getParent();  
 **currentTabTag** = tabHost.getCurrentTabTag();  
 View view = **null**;  
  
 Intent intent = getActivity().getIntent();  
 *// retrieve the username from the intent just passed from the MainActivity* String username = intent.getStringExtra(**"username"**);  
 **int** userID = intent.getIntExtra(**"userID"**, -1);  
  
 User searchedUser = MainActivity.*pickersDB*.getUser(userID);  
  
 *// here I created three different tabs* **if** (**currentTabTag**.equalsIgnoreCase(**"Profile"**)) *//when users select the Profile tab* {  
 view = inflater.inflate(R.layout.***fragment\_profile\_info***, container, **false**);  
  
 **user\_name\_txtView** = (TextView) view.findViewById(R.id.***username\_info\_txtView***);  
 **user\_firstName\_txtView** = (TextView) view.findViewById(R.id.***firtname\_info\_txtView***);  
 **user\_lastName\_txtView** = (TextView) view.findViewById(R.id.***lastname\_info\_txtView***);  
 **user\_password\_txtView** = (TextView) view.findViewById(R.id.***password\_info\_txtView***);  
 **user\_email\_txtView** = (TextView) view.findViewById(R.id.***email\_info\_txtView***);  
 **user\_dob\_txtView** = (TextView) view.findViewById(R.id.***dob\_info\_txtView***);  
 **user\_intro\_txtView** = (TextView) view.findViewById(R.id.***introduction\_info\_txtView***);  
  
 **user\_name\_txtView**.setText(searchedUser.getUserName());  
 **user\_firstName\_txtView**.setText(searchedUser.getFirstName());  
 **user\_lastName\_txtView**.setText(searchedUser.getLastName());  
 **user\_password\_txtView**.setText(searchedUser.getPassword());  
 **user\_email\_txtView**.setText(searchedUser.getEmail());  
 **user\_dob\_txtView**.setText(searchedUser.getDateOfBirth());  
 **user\_intro\_txtView**.setText(searchedUser.getIntroduction());  
  
 **functionTextView** = (TextView) view.findViewById(R.id.***function\_textView***);  
  
  
 } **else if** (**currentTabTag**.equalsIgnoreCase(**"Search"**)) *//when users select the Search tab* {  
 view = inflater.inflate(R.layout.***fragment\_map***, container, **false**);  
  
 **show\_on\_map\_btn** = (Button) view.findViewById(R.id.***show\_on\_map\_btn***);  
 **show\_on\_map\_btn**.setOnClickListener(**this**);  
  
 } **else if** (**currentTabTag**.equalsIgnoreCase(**"Contacts"**)) *//when users select the Contacts tab* {  
 view = inflater.inflate(R.layout.***fragment\_contacts***, container, **false**);  
  
 String[] columns = {  
 Data.***\_ID***, *// primary key* Contacts.***DISPLAY\_NAME***, *// person's name* Data.***DATA1***, *// phone number* Data.***DATA2*** *// phone type (mobile, home, work, etc.)* };  
  
 *//String where = "("+Data.MIMETYPE + "='"+Phone.CONTENT\_ITEM\_TYPE+"')";  
 //String orderBy = Contacts.TIMES\_CONTACTED+" DESC";* Cursor cursor = getActivity().getContentResolver().query(**DATA\_URI**, columns, **null**, **null**, **null**);  
 ArrayList<User> contactList = **new** ArrayList<User>();  
  
 **while** (cursor.moveToNext()) *//keeps going through all the contacts until it reaches the end* {  
 User user = **new** User();  
 user.setUserName(cursor.getString(1));  
  
 contactList.add(user); *// add each User object to the ArrayList object connactList for later displaying in a ListView* }  
  
  
 ArrayList<HashMap<String, String>> data = **new** ArrayList<>();  
  
 **for** (User user : contactList) {  
 HashMap<String, String> map = **new** HashMap<String, String>();  
 map.put(**"contact\_name"**, user.getUserName());  
 map.put(**"contact\_email"**, user.getEmail());  
 data.add(map);  
 }  
  
 **int** resource = R.layout.***contacts\_listview\_item***;  
 String[] from = {**"contact\_name"**, **"contact\_email"**};  
 **int**[] to = {R.id.***contact\_name***, R.id.***contact\_email***};  
  
 SimpleAdapter adapter = **new** SimpleAdapter(getContext(), data, resource, from, to);  
 **contact\_list\_view** = (ListView) view.findViewById(R.id.***contactsListView***);  
 **contact\_list\_view**.setAdapter(adapter);  
  
 }  
  
  
 *//refreshFragment();* **return** view;  
 }  
  
 */\*  
 public void refreshFragment() {  
 String text = "This is the " + currentTabTag;  
 //functionTextView.setText(text);  
 }  
 \*/* @Override  
 **public void** onResume() {  
 **super**.onResume();  
 *//refreshFragment();* }  
  
  
 @Override  
 **public void** onClick(View v) {  
  
 **switch** (v.getId())  
 {  
 **case** R.id.***show\_on\_map\_btn***:  
 String link = **"https://www.google.com/maps"**;  
 Uri uri = Uri.*parse*(link);  
 *// send a VIEW action to the target uri* Intent viewIntent = **new** Intent(Intent.***ACTION\_VIEW***, uri);  
 startActivity(viewIntent);  
 **break**;  
 }  
  
 }  
}

# Below is the matrix that will be used to evaluate your response:



A regular A translates as 95, A-=93, B+=87, B=85, B-=83, C+=77, C=75, C-=73, D+=67, F=0 etc. To get an A grade for the course, your weighted average should be >93. A-:>=90. B+:>=87. B:>83. B-:>=80. C+:>=77. C:>73. C-:>=70 etc.

1. Document intentions—don’t paraphrase code. Nontrivial functions should have (an informal) *Intent* statement*,* (precise) *Preconditions* (if any), *Returns* (if any),and *Postconditions* (always). Each block of code should be preceded by its intended objectives. [↑](#footnote-ref-1)