In-Class Homework #10 – Useful Third Party Libraries for Android

CUS 357 – Winter 2017
Due Date: End of class, Mar 31, 2017

Learning Objectives

- Learn about ButterKnife and Parceler – two handy libraries for Android.
- Become more proficient at researching and utilizing Android platform features on your own.

Please work in pairs to get as much of this assignment done as possible by the end of today's class session.

Overview

Begin this exercise using the finished product of your In-Class Homework #7 solution. If you did not finish that homework, you are welcome to start with the instructor's solution, available on Blackboard.

You are to bring your Android GeoCalculator implementation up to par with the iOS version. That is, add a new activity (LocationSearchActivity) that simplifies the entry of geo coordinates, as shown in Figure 1 below. Along the way, you are to utilize the ButterKnife and Parceler as demonstrated in Lecture 13 (part 02) in your LocationSearchActivity implementation. For the Date entry on the location form, utilize the native Android DatePickerDialog component.

Your implementation in GeoCalculator will be similar to the Android Traxy code in Chapter 07 of the text. So similar that instead of spoon feeding you the step by step details, you're provided with the instructor's demo notes for the Android code in Chapter 07. If you've understood the text / lecture materials it should be straightforward to re-purpose the instructions/code for this assignment.

There are a few differences to be aware of:

- You will have two location fields on your form, so you will have to key each start of the Google Places Picker with a different value so you can distinguish the callbacks in your onActivityResult method.
- You will only have one date field. You will use the simple Android DatePickerDialog instead of the fancy third party version used in Traxy. (Note the date is not used at this point, you're just getting some experiences exploring Android.)
- You will introduce a new Parcelable class LocationLookup instead of Trip.

```
@Parcel
public class LocationLookup {
    double origLat;
    double origLng;
    double endLat;
    double endLng;
}
```

If you finish before the end of the session, give a demo to the instructor.

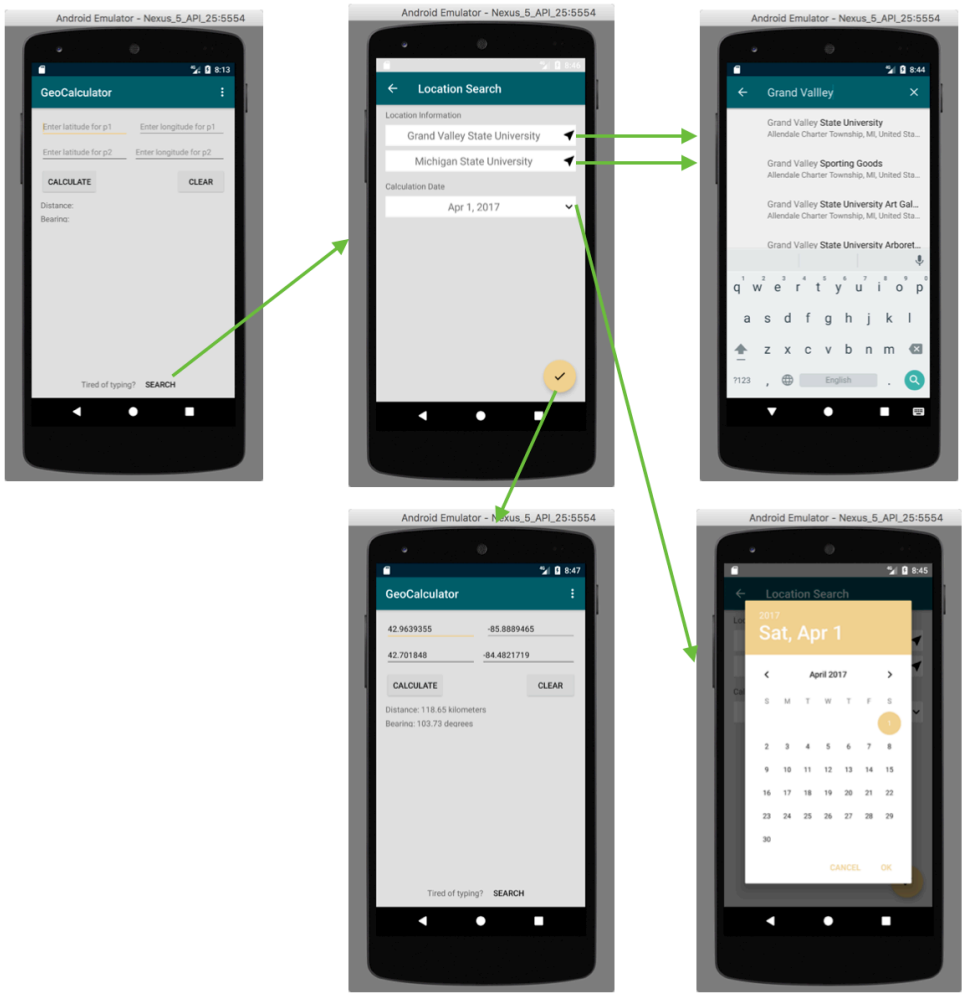

Figure 1. Adding LocationSearchActivity to the Android GeoCalculator.

References

Source code for the textbook: https://github.com/gvsucis/mobile-app-dev-book
DatePickerDialog - https://developer.android.com/guide/topics/ui/controls/pickers.html
Parceler docs - https://github.com/johncarl81/parceler
ButterKnife docs - http://jakewharton.github.io/butterknife/
Google Places API for Android - https://developers.google.com/places/android-api/

Traxy - Working with Third Party Code - Android Section of Ch07

Happily, Android Studio has built-in support for dependency management using gradle. You're already familiar with gradle, so we don't need to spend time on it now.

However, there are lots of third party components that can make our life a lot easier. Let's take a look at a few useful components, and demonstrate how they can be used to bring our Android implementation Traxy up to par with our iOS.

Recall, we want to add a new screen to allow the creation of new journals.

1. Begin by adding the dependencies we will use to your gradle build:

   ```
   compile 'com.google.android.gms:play-services-places:10.2.1'
   compile 'com.borax12.materialdaterangepicker:library:1.9'
   compile 'org.parceler:parceler-api:1.1.6'
   annotationProcessor 'org.parceler:parceler:1.1.6'
   compile 'com.jakewharton:butterknife:8.4.0'
   annotationProcessor 'com.jakewharton:butterknife-compiler:8.4.0'
   ```

   You are already familiar with google places. MaterialDateRangePicker let's us pick a range of dates in a single dialog. The ParcelerAPI let's us easily add parcelizing logic to any old Java class with a simple annotation. ButterKnife gives us an elegant solution to linking our Java code to identifiers in our layout files.

2. Use New > Activity to create a new Basic Activity called NewJournalActivity. Set its title to "New Journal", and set "MainActivity" as its hierarchical parent.

3. Set the content of content*new*journal.xml layout to the following:

   ```
   <?xml version="1.0" encoding="utf-8"?>
   <android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/and
   roid"
       xmlns:app="http://schemas.android.com/apk/res-auto"
       xmlns:tools="http://schemas.android.com/tools"
       android:layout_width="match_parent"
       android:layout_height="match_parent"
       app:layout_behavior="@string/appbar_scrolling_view_behavior"
       tools:context="cis.gvsu.edu.traxy.NewJournalActivity"
       tools:showIn="@layout/activity_new_journal">
       <EditText
           android:id="@+id/journal_name"
           android:layout_width="0dp"
           android:layout_height="36dp"
           android:layout_marginEnd="16dp"
           android:layout_marginStart="16dp"
           android:layout_marginTop="8dp"
           android:background="@android:color/white"
           android:ems="10"
   ```

```
        android:gravity="center"
        android:hint="@string/journal_name"
        android:inputType="textPersonName"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView"
        tools:layout_constraintLeft_creator="1"
        tools:layout_constraintRight_creator="1" />

    <TextView
        android:id="@+id/location"
        android:layout_width="0dp"
        android:layout_height="36dp"
        android:layout_marginEnd="16dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="8dp"
        android:background="@android:color/white"
        android:drawableEnd="@drawable/ic_near_me_black_24dp"
        android:gravity="center"
        android:hint="@string/location_name"
        android:textAppearance="@style/TextAppearance.AppCompat.Medium"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/journal_name" />

    <TextView
        android:id="@+id/start_date"
        android:layout_width="0dp"
        android:layout_height="36dp"
        android:layout_marginTop="8dp"
        android:background="@android:color/white"
        android:drawableEnd="@drawable/ic_expand_more_black_24dp"
        android:gravity="center"
        android:hint="@string/start_date_hint"
        android:textAppearance="@style/TextAppearance.AppCompat.Medium"
        app:layout_constraintLeft_toLeftOf="@+id/journal_name"
        app:layout_constraintRight_toRightOf="@+id/journal_name"
        app:layout_constraintTop_toBottomOf="@+id/textView2" />

    <TextView
        android:id="@+id/end_date"
        android:layout_width="0dp"
        android:layout_height="36dp"
        android:layout_marginTop="8dp"
        android:background="@android:color/white"
        android:drawableEnd="@drawable/ic_expand_more_black_24dp"
        android:gravity="center"
        android:hint="@string/end_date_hint"
```

```
        android:textAppearance="@style/TextAppearance.AppCompat.Medium"
        app:layout_constraintLeft_toLeftOf="@+id/start_date"
        app:layout_constraintRight_toRightOf="@+id/start_date"
        app:layout_constraintTop_toBottomOf="@+id/start_date" />
    <TextView
        android:text="@string/trip_info"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/textView"
        android:layout_marginStart="16dp"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        android:layout_marginTop="8dp"
        />
    <TextView
        android:text="@string/trip_interval"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/textView2"
        android:layout_marginStart="16dp"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/location"
        android:layout_marginTop="16dp"
        />
</android.support.constraint.ConstraintLayout>
```

4. The strings referenced in our layout are defined as follows (e.g. places these in res/values/strings.xml):

```
<string name="trip_info">Trip Information</string>
<string name="journal_name">Enter Journal Name</string>
<string name="location_name">Location Name</string>
<string name="start_date_hint">Start Date</string>
<string name="end_date_hint">End Date</string>
<string name="trip_interval">Trip Date</string>
```

5. Make the vector images for ADD and CHECK a priori, set them on the activity*main.xml and activity*new_journal.xml

6. Add the remaining images referenced in the layout (e.g. "near me" icon and "expand more") as Vector images. See the demo in Lecture 13, Part 02 if you've forgotten how to do this.

7. Make MainActivity set up the google client by placing this call in its onCreate method:

```
GoogleApiClient apiClient = new GoogleApiClient.Builder(this)
    .addApi(Places.GEO_DATA_API)
    .addApi(Places.PLACE_DETECTION_API)
    .enableAutoManage(this, this)
    .build();
```

Note that you will get a compile error on the second parameter of enableAutoManage - make sure MainActivity implements the missing protocol and it will go away. You can simply put a Log statement in the method or leave it empty for now.

8. Update your manifest file to point to your Google API key. In particular, add this element to the element, replacing the YOUR-KEY-GOES-HERE string with your actual key value.

```
<meta-data
    android:name="com.google.android.geo.API_KEY"
    android:value="YOUR-KEY-GOES-HERE" />
```

9. Add a new class to serve as your model:

```
@Parcel
public class Trip {
    String name, location, placeId;
    String startDate, endDate;
    double lat, lng;
}
```

By adding this simple annotation we make it parcelizable. Consult the Android documentation for more information on exactly what that means.

10. Add the following implementation for AddNewJournalActivity. This is the complete implementation of the class. Review it carefully and make sure you understand the code.

```
package cis.gvsu.edu.traxy;

import android.content.Intent;
import android.os.Bundle;
import android.os.Parcelable;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.util.Log;
import android.widget.EditText;
import android.widget.TextView;
import com.borax12.materialdaterangepicker.date.DatePickerDialog;
import com.google.android.gms.common.GooglePlayServicesNotAvailableException;
import com.google.android.gms.common.GooglePlayServicesRepairableException;
import com.google.android.gms.common.api.Status;
import com.google.android.gms.location.places.Place;
import com.google.android.gms.location.places.ui.PlaceAutocomplete;
import org.joda.time.DateTime;
import org.joda.time.format.DateTimeFormatter;
import org.joda.time.format.ISODateTimeFormat;
import org.parceler.Parcels;
import java.util.Locale;
import butterknife.BindView;
```

```
import butterknife.ButterKnife;
import butterknife.OnClick;

public class NewJournalActivity extends AppCompatActivity implements DatePickerDialog.OnDateSetLis
tener {

    int PLACE_AUTOCOMPLETE_REQUEST_CODE = 1;
    private static final String TAG = "NewJournalActivity";

    @BindView(R.id.journal_name) EditText jname;
    @BindView(R.id.location) TextView location;
    @BindView(R.id.start_date) TextView startDateView;
    @BindView(R.id.end_date) TextView endDateView;

    private DateTime startDate, endDate;
    private DatePickerDialog dpDialog;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_new_journal);
        ButterKnife.bind(this);

        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
        getSupportActionBar().setDisplayHomeAsUpEnabled(true);
        DateTime today = DateTime.now();
        dpDialog = DatePickerDialog.newInstance(this,
                today.getYear(), today.getMonthOfYear() - 1, today.getDayOfMonth());

        startDateView.setText(formatted(today));
        endDateView.setText(formatted(today.plusDays(1)));
    }

    @OnClick(R.id.location)
    public void locationPressed() {
        try {
            Intent intent =
                    new PlaceAutocomplete.IntentBuilder(PlaceAutocomplete.MODE_FULLSCREEN)
                            .build(this);
            startActivityForResult(intent, PLACE_AUTOCOMPLETE_REQUEST_CODE);
        } catch (GooglePlayServicesRepairableException e) {
            e.printStackTrace();
        } catch (GooglePlayServicesNotAvailableException e) {
            e.printStackTrace();
        }
    }
```

```
    @OnClick({R.id.start_date, R.id.end_date})
    public void datePressed() {
        dpDialog.show(getFragmentManager(), "daterangedialog");
    }

    @OnClick(R.id.fab)
    public void FABPressed() {
        Intent result = new Intent();
        Trip aTrip = new Trip();
        aTrip.name = jname.getText().toString();
        aTrip.location = location.getText().toString();
        DateTimeFormatter fmt = ISODateTimeFormat.dateTime();
        aTrip.startDate = fmt.print(startDate);
        aTrip.endDate = fmt.print(endDate);
        // add more code to initialize the rest of the fields
        Parcelable parcel = Parcels.wrap(aTrip);
        result.putExtra("TRIP", parcel);
        setResult(RESULT_OK, result);
        finish();
    }

    private String formatted(DateTime d) {
        return d.monthOfYear().getAsShortText(Locale.getDefault()) + " " +
                d.getDayOfMonth() + ", " + d.getYear();
    }
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        if (requestCode == PLACE_AUTOCOMPLETE_REQUEST_CODE) {
            if (resultCode == RESULT_OK) {
                Place pl = PlaceAutocomplete.getPlace(this, data);
                location.setText(pl.getAddress());
                Log.i(TAG, "onActivityResult: " + pl.getName() + "/" + pl.getAddress());

            } else if (resultCode == PlaceAutocomplete.RESULT_ERROR) {
                Status stat = PlaceAutocomplete.getStatus(this, data);
                Log.d(TAG, "onActivityResult: ");
            }
            else if (requestCode == RESULT_CANCELED){
                System.out.println("Cancelled by the user");
            }
        }
        else
            super.onActivityResult(requestCode, resultCode, data);
    }

    @Override
    public void onDateSet(DatePickerDialog view, int year, int monthOfYear, int dayOfMonth, int ye
arEnd, int monthOfYearEnd, int dayOfMonthEnd) {
```

```
        startDate = new DateTime(year, monthOfYear + 1, dayOfMonth, 0, 0);
        endDate = new DateTime(yearEnd, monthOfYearEnd + 1, dayOfMonthEnd, 0, 0);
        startDateView.setText(formatted(startDate));
        endDateView.setText(formatted(endDate));
    }
}
```

11. Now let's modify the main activity so its FAB starts up our NewJournalActivity:

```
fab.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Intent newJournal = new Intent(MainActivity.this, NewJournalActivity.class);
        startActivityForResult(newJournal, NEW_TRIP_REQUEST);
    }
});
```

12. And finally, Since we've flattened our new trip object out as a parcelable when adding, we need to adjust the code in MainActivity to receive and unflatten it. In onActivityResult make these mods:

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == NEW_TRIP_REQUEST) {
        if (data != null && data.hasExtra("TRIP")) {
            Parcelable parcel = data.getParcelableExtra("TRIP");
            Trip t = Parcels.unwrap(parcel);
            Log.d("MainACtivity", "New Trip: " + t.name);
        }
    }
    else
        super.onActivityResult(requestCode, resultCode, data);
}
```

13. Now go ahead and run your app and give it a try. Congratulations on a job well done!