Summary file for summarizing the steps to creating individual items making the project:

* Activities and Fragments:

1. Set parent/child activity/fragment in manifest file
2. Set layout of the Activity in layout XML file to the fragment layout
3. Activity and Fragents must have Layout XML file

tools:layout="@layout/fragment\_main"

1. In Fragment “” create a View object to address the fragment View, since there’s no method for getting rootView for fragments

View rootView = inflater.inflate(R.layout.*fragment\_main*, container, false);

1. Use “onStart()” to run code at the start of an Activity/Fragment
2. Use “getActivity()” to get the parent Activity of the Fragment to call methods that work on the Activity

* Starting an Activity:

1. Create Intent for Activity to be started  
   Intent settingsIntent = new Intent(this, SettingsActivity.class);
2. Set data to Activity if needed  
   intent.setData(geoLocation);
3. Start the Activity  
   startActivity(settingsIntent);

* Adding OptionsMenu to:
* Activity:

1. Create a menu XML file with <menu> and <item> tags.
2. In “onCreateOptionsMenu” add “getMenuInflater().inflate(R.menu.menu\_name, menu);”
3. Handle menu/action bar item selection in “onOptionsItemSelected”

public boolean onOptionsItemSelected(MenuItem item) {  
 // Handle item selection  
 int id = item.getItemId();  
 switch (id) {  
 case R.id.*action\_refresh*:  
 //Do refresh action  
 updateWeather();  
 return true;  
 default:  
 return super.onOptionsItemSelected(item);  
 }  
}

* Fragment:

1. In “onCreate” method add

setHasOptionsMenu(true); //Indicates that this fragment contributes to the menu and so calls "onCreateOptionsMenu"

1. Create a menu XML file with <menu> and <item> tags.
2. In “onCreateOptionsMenu” add “getMenuInflater().inflate(R.menu.menu\_name, menu);”
3. Handle menu/action bar item selection in “onOptionsItemSelected”

public boolean onOptionsItemSelected(MenuItem item) {  
 // Handle item selection  
 int id = item.getItemId();  
 switch (id) {  
 case R.id.*action\_refresh*:  
 //Do refresh action  
 updateWeather();  
 return true;  
 default:  
 return super.onOptionsItemSelected(item);  
 }  
}

* Adapters:

1. Create an Adapter for certain Views that require data updating

mForecastAdapter = new ArrayAdapter<>(  
 getActivity(), // The current context (this activity)  
 R.layout.*list\_item\_forecast*, // ID of list item layout  
 R.id.*list\_item\_forecast\_textview*, // The ID of the textview to populate.  
 new ArrayList<String>());

1. Create a specific type of View object

ListView forecastListView = (ListView)rootView.findViewById(R.id.*listview\_forecast*);

1. Set the Adapter to the View

forecastListView.setAdapter(mForecastAdapter);

1. Deal with View clicking by adding Lister:

forecastListView.setOnItemClickListener(new AdapterView.OnItemClickListener() {  
 @Override  
 public void onItemClick(AdapterView<?> adapterView, View view, int position, long l) {  
 //Make message appear as a toast  
 Context context = getActivity();//.getApplicationContext();  
 CharSequence text = (String) adapterView.getItemAtPosition(position);  
 CharSequence text1 = mForecastAdapter.getItem(position); //This is used in the lesson  
 int duration = Toast.*LENGTH\_SHORT*;  
 Toast toast = Toast.*makeText*(context, text, duration);  
 toast.show();  
  
 String forecast = (String) adapterView.getItemAtPosition(position);  
 Intent intent = new Intent(getActivity(), DetailActivity.class);  
 intent.putExtra(Intent.*EXTRA\_TEXT*, forecast);  
 startActivity(intent);  
 }

* Logging:

1. Create a log tag
2. Use “Log” class (Log.d(LOG\_TAG string, string to log))

* Settings (Shared preferences)
* To get a setting:

1. Get SharedPreferences class  
   SharedPreferences sharedPref = PreferenceManager.*getDefaultSharedPreferences*(this);
2. Get the settings String  
   String location = sharedPref.getString(  
    getString(R.string.*pref\_location\_key*), //Get the value stored in this key  
    getString(R.string.*pref\_location\_default*)); //Return this default value if value doesn't exist

* Opening a Geo location in another App:

//Get preferred location from SharedPreferences / settings  
SharedPreferences sharedPref = PreferenceManager.*getDefaultSharedPreferences*(this);  
String location = sharedPref.getString(  
 getString(R.string.*pref\_location\_key*), //Get the value stored in this key  
 getString(R.string.*pref\_location\_default*)); //Return this default value if value doesn't exist  
  
//Making Uri for geo location intent  
Uri geoLocation = Uri.*parse*("geo:0,0?").buildUpon()  
 .appendQueryParameter("q", location)  
 .build();  
Intent intent = new Intent(Intent.*ACTION\_VIEW*);  
intent.setData(geoLocation);  
  
if (intent.resolveActivity(getPackageManager()) != null) { //Calling an app that might not exist  
 startActivity(intent);  
} else {  
 Log.*d*(LOG\_TAG, "Couldn't call " + location + ", no receiving apps installed!");  
}