Special Interest Group on Innovative Software 2012-2013
Workshop on Android app development

Session 3: Android App Development – App ←→Server Communication and Location Service

Contents

Prepared by Mr. Felix Tam, Committee Member of the Special Interest Group (SIG) on Innovative Software 2012-2013, Department of Computer Science, Hong Kong Baptist University.

0 Before you start

Please follow the instruction written in part 1 of our session 1 note to setup the development environment. You may get the session 1 note from http://www.comp.hkbu.edu.hk/~sigis/android/2013-14/day1.pdf

In short, you need to have the following three tools ready:

1. Eclipse IDE with built-in ADT (Android Developer Tools)
   http://developer.android.com/sdk/index.html

2. JDK 1.7(Windows 64bit)
   http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html

3. Android emulator or your Android device with USB debugging option enabled

---

Alternatively, you may download the tools.zip from the share drive which contains a pre-configured Eclipse w/ADT and JDK 1.7. Extract the tools.zip and put the **"adt-bundle-windows-x86_64-20131030"** folder into **"D:\"**. Remember to double click the **"jdk-7u51-windows-x64.exe"** to install the JDK 1.7 before you start Eclipse.

The default workspace location for the pre-configured Eclipse w/ADT is **"D:\adt-bundle-windows-x86_64-20131030\eclipse\workspace"**.

In this session, we will use "learn by example" approach to go through our topics. You are suggested to copy the codes directly and further explanation will be given during the workshop.

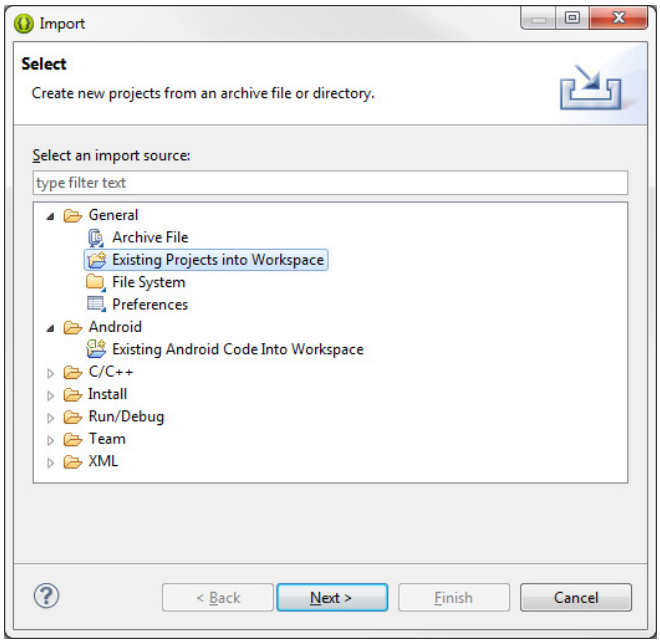
1 Detect location and manage Maps objects

1.1 Detect user current location

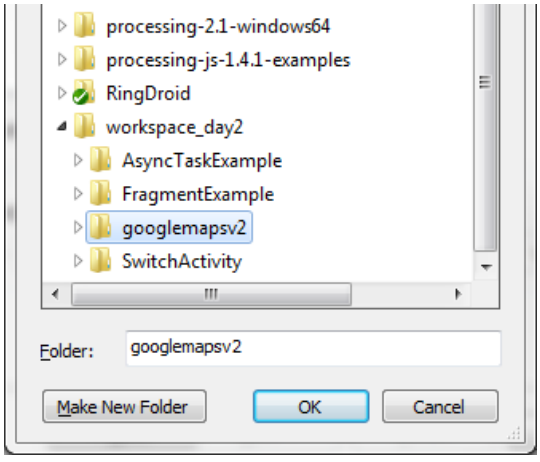
1. We will base on the example **com.example.googlemapsv2** that we created in session 2. Download the workspace_day2.zip from http://www.comp.hkbu.edu.hk/~sigis/android/2013-14/workspace_day2.zip.

2. Import the Google Play services as a library project in Eclipse. For details, please refer to session 2 notes (Part 4.1).

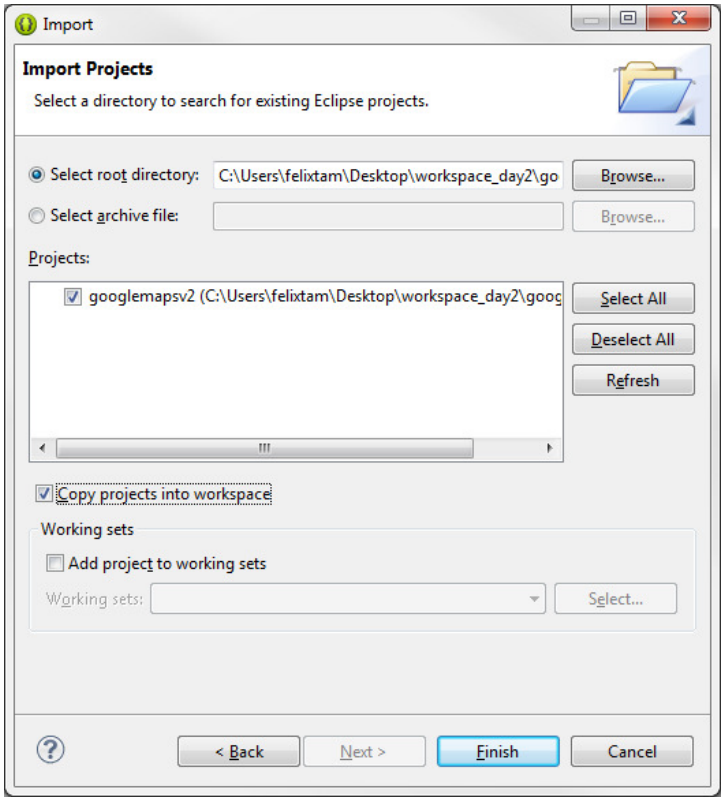
3. Extract the zip file and go to Eclipse, click **File** > **Import** > **Existing Projects into Workspace**.



4. Click **Browse,** locate the folder called **googlemapsv2** which extracted in step 1.



5. Check the box **Copy projects into workspace** and click **Finish**.



6. Replace the contents in **MainActivity.java** with the following codes:

```
package com.example.googlemapsv2;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.CameraPosition;
import com.google.android.gms.maps.model.CameraPosition.Builder;
import com.google.android.gms.maps.model.LatLng;

import android.content.Context;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.support.v4.app.FragmentActivity;

public class MainActivity extends FragmentActivity {
```

```
        private GoogleMap googleMap;
        private LocationManager locationManager;
        private LocationListener locationListener;
        private Builder cameraPositionBuilder;
        private CameraPosition cameraPosition;

        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);

            googleMap = ((SupportMapFragment)
getSupportFragmentManager().findFragmentById(R.id.map)).getMap();

            cameraPositionBuilder = new CameraPosition.Builder();
            cameraPositionBuilder.zoom(17);

            locationManager = (LocationManager)
this.getSystemService(Context.LOCATION_SERVICE);
            locationListener = new LocationListener() {
                @Override
                public void onLocationChanged(Location location) {
                    cameraPositionBuilder.target(new
LatLng(location.getLatitude(), location.getLongitude()));
                    cameraPosition = cameraPositionBuilder.build();

        googleMap.animateCamera(CameraUpdateFactory.newCameraPosition(
cameraPosition));
                }

                @Override
                public void onStatusChanged(String provider, int status, Bundle
extras) {}

                @Override
                public void onProviderEnabled(String provider) {}

                @Override
```

```
                public void onProviderDisabled(String provider) {}

            };
            locationManager.requestLocationUpdates(
 LocationManager.NETWORK_PROVIDER, 0, 0, locationListener);

 locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0,
 0, locationListener);
        }
}
```

7. Copy the **debug.keystore** (backed up from session 2) and overwrite the existing file (if any) in **C:\Users\<your_name>\.android** and **C:\Users\test1\.android\.**

8. Replace the map API key in AndroidManifest.xml with the one created in session 2.

```
 <meta-data
            android:name="com.google.android.maps.v2.API_KEY"
            android:value="your_api_key"/>
```

Note: If you lost your debug.keystore file created in the previous session, please follow the steps written down in session 2 notes (Part 4) again.

Code explanation:

- We use **(SupportMapFragment) getSupportFragmentManager().findFragmentById(R.id.map)).getMap()** to get the map from the map fragment.
- We use a **Builder** object to build a camera view which contains target, zoom, bearing and tilt information.
- We use the **LocationManager** to get the latitude and longitude from the device.
- We use the **LocationListener** to listen for the change of latitude and longitude.
- When the app acquires the location, it will use the **Builder** to build the **CameraPosition**. Afterwards, we will use the **CameraUpdateFactory** to construct the **CameraUpdate** object and supply it to **animateCamera()** of **GoogleMap.**

The following markups in **AndroidManifest.xml** (above <application> tag) are used to declare the permission of location access.

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"
/>
<uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION" />
```

Note:

ACCESS_FINE_LOCATION : Access Precise location from GPS

ACCESS_ COARSE_LOCATION : Access Approximate location from Wi-Fi and cellular.

9. Now test the app in the device and remember to activate your internet connection. You will soon see the map and the view will be moved to your current location after a while.

1.2 Manage Maps objects

You can add different objects to Google Maps including markers, polylines and

polygons. In our workshop, we will focus on how to add a default marker and customize it later on. We will continue to base on the example **com.example.googlemapsv2** that we just completed in section 1.1.

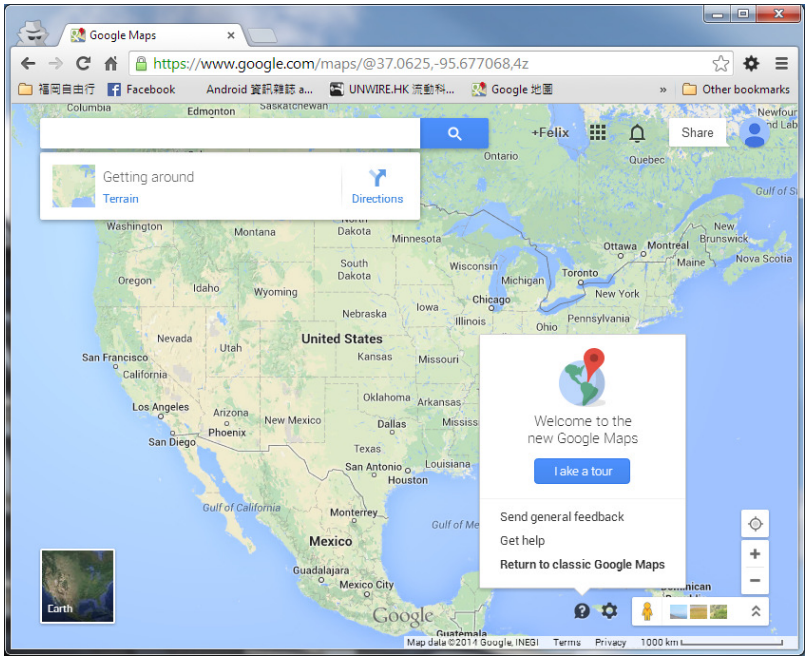
Note: Import any necessary packages using Eclipse Quick-fix feature when you are prompted for errors due to un-resolved classes.

1. Inside the **onCreate(),** add the following codes after getting the Google Maps instance.

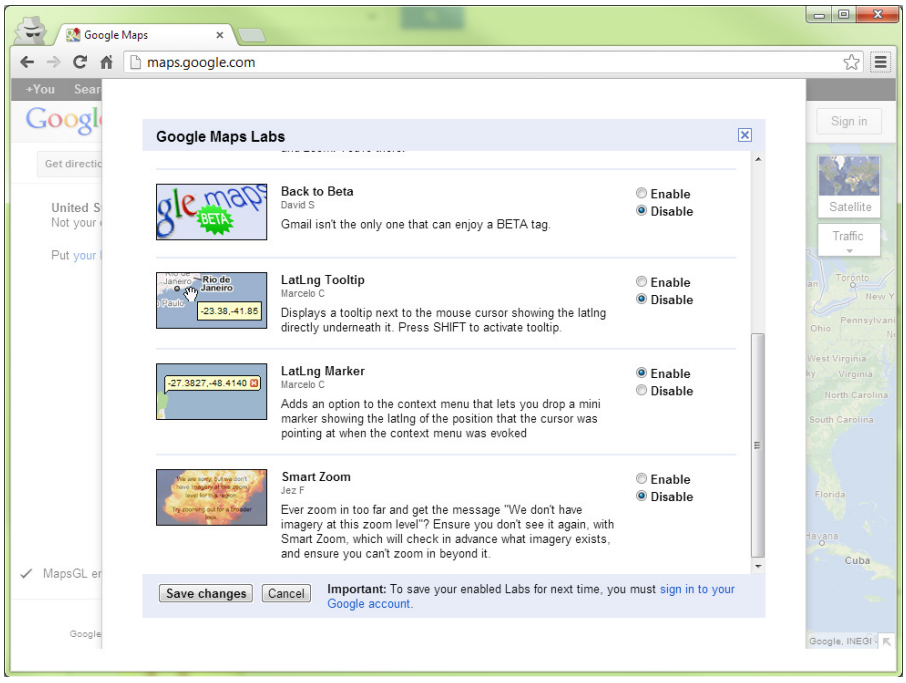
   ```
   MarkerOptions markerOptions = new MarkerOptions();
   markerOptions.position(new LatLng(22.340405, 114.179580));
   markerOptions.title("RRS 638, Sir Run Run Shaw Building HKBU");
   googleMap.addMarker(markerOptions);
   ```

2. Now run the app and you will get a default marker placed on our campus. You can click on the marker and see the title of the marker.

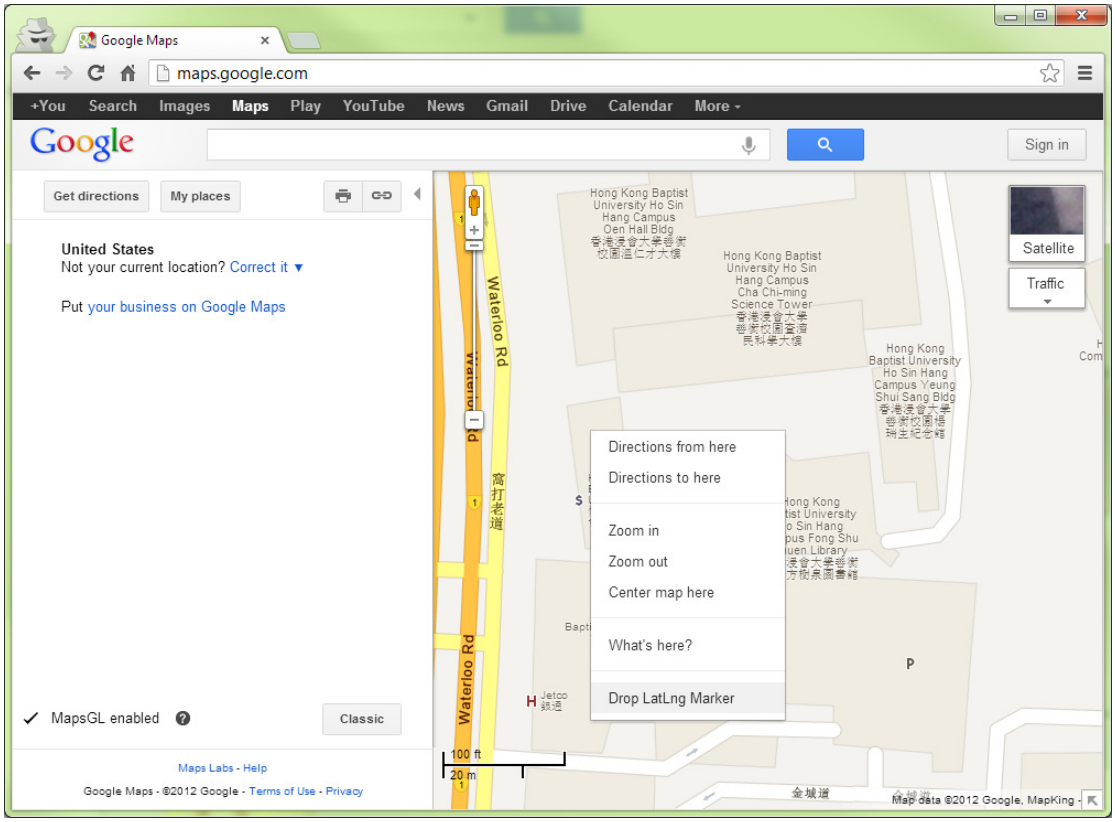
3. To get the latitude and longitude of a specific location, go to http://maps.google.com/. However, we need to use the classic Google Maps instead. To switch from the new version to the classic version, click on the **?** icon at the bottom right corner and choose **Return to classic Google Maps**



4. Click on **Maps Labs** at the bottom left corner. Find out **LatLng Marker** and enable it. Click **Save changes** to proceed.



5. Now you can right click on the map anywhere and choose **Drop LatLng Marker**.



6. You will get a balloon with latitude and longitude information. Copy and paste them when creating a new **LatLng** object if necessary.

7. In the following steps, we will change the default marker to our department logo and supply a snippet to it. Download the department logo from

http://www.comp.hkbu.edu.hk/~sigis/android/public/csd_logo.png

8. Copy the image to **drawable-hdpi** folder of your project and set the icon for the **markerOptions** object using the following code:

   markerOptions.icon(BitmapDescriptorFactory.fromResource(
   R.drawable.csd_logo));

9. Also, use the **snippet()** method to set the snippet of the **markerOptions** object. For example: markerOptions.snippet("Department of Computer Science");
   Note: You should call the icon() as well as snippet() before adding the marker using the addMarker() method of GoogleMap.

10. Run the app and you should get the following result.

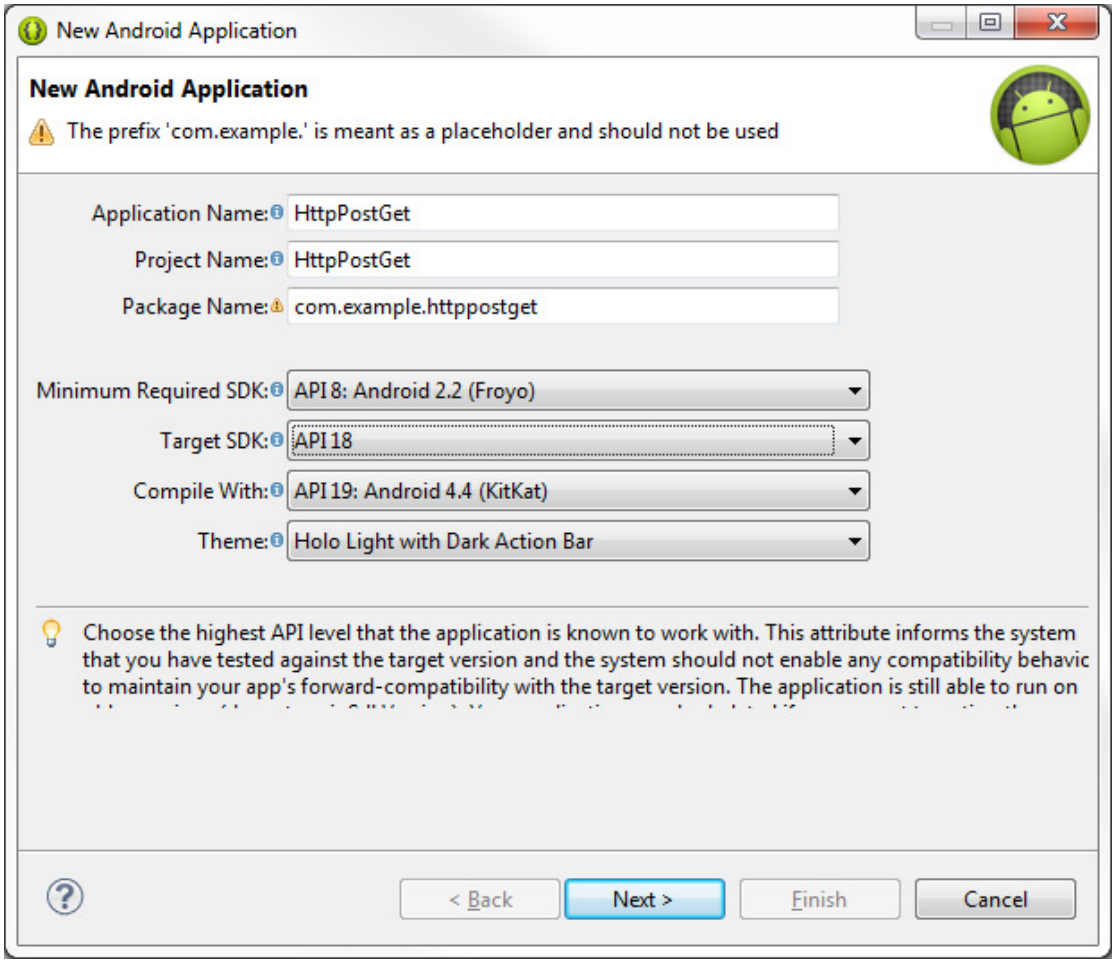
2 Communicate with server

To communicate with server using an Android device, we may make use of either HTTP POST or GET to send data and retrieve data in JSON format.

2.1 Send and receive data using HTTP POST/GET

1. In this tutorial, we have a PHP script (http://www.comp.hkbu.edu.hk/~sigis/android/public/handler.php) which accepts a string and then returns the name followed by current date and time. The variable name of the string for POST and GET are post_string and get_string respectively. The source code of the PHP script is included at the end of this section.

2. Create a new project called **HttpPostGet** with **Create activity** option checked in **Configure Project** step**.** Click next in the rest of steps and refer to the screenshot below for the information at the first step.



3. Replace the contents in **activity_main.xml** with the following markups:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <EditText
        android:id="@+id/userInput"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal" >
        <Button
            android:id="@+id/postSend"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Send w/ POST"
            android:layout_weight="1"/>
        <Button
            android:id="@+id/getSend"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Send w/ GET"
            android:layout_weight="1"/>
    </LinearLayout>
    <TextView
        android:id="@+id/result"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

4. Replace the contents in **MainActivity.java** with the following codes:

```
package com.example.httppostget;
import java.net.URLEncoder;
import java.util.ArrayList;
```

```
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.apache.http.protocol.HTTP;
import org.apache.http.util.EntityUtils;

import android.app.Activity;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;

public class MainActivity extends Activity {

    private Button sendWithPostBtn;
    private Button sendWithGetBtn;
    private TextView inputTv;
    private TextView resultTv;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        sendWithPostBtn = (Button)findViewById(R.id.postSend);
        sendWithGetBtn = (Button)findViewById(R.id.getSend);
        inputTv = (TextView)findViewById(R.id.userInput);
        resultTv = (TextView)findViewById(R.id.result);

        sendWithPostBtn.setOnClickListener(sendWithPostBtnListener);
        sendWithGetBtn.setOnClickListener(sendWithGetBtnListener);
```

```
	}

	private OnClickListener sendWithPostBtnListener = new OnClickListener(){
		@Override
		public void onClick(View arg0) {
			new SentWithAsyncTask().execute("POST");
		}
	};

	private OnClickListener sendWithGetBtnListener = new OnClickListener(){
		@Override
		public void onClick(View arg0) {
			new SentWithAsyncTask().execute("GET");
		}
	};

	private String sendWtihPost(String inputStr){
		HttpPost request = new
HttpPost("http://www.comp.hkbu.edu.hk/~sigis/android/public/handler.php");
		ArrayList<NameValuePair> params = new ArrayList<NameValuePair>();
		params.add(new BasicNameValuePair("post_string", inputStr));
		String result = "";
		try{
			request.setEntity(new UrlEncodedFormEntity(params,
HTTP.UTF_8));
			HttpResponse httpResponse = new
DefaultHttpClient().execute(request);
			if(httpResponse.getStatusLine().getStatusCode() == 200){
				result = EntityUtils.toString(httpResponse.getEntity());
			}
		}catch(Exception e){
			e.printStackTrace();
		}
		return result;
	}

	private String sendWtihGet(String inputStr){
		String result = "";
```

```
            try {
                HttpGet request = new HttpGet(
"http://www.comp.hkbu.edu.hk/~sigis/android/public/handler.php?get_string=" +
URLEncoder.encode(inputStr, "UTF-8"));
                HttpResponse response = new
DefaultHttpClient().execute(request);
                result = EntityUtils.toString(response.getEntity());
            }catch(Exception e) {
                e.printStackTrace();
            }
            return result;
        }

        private class SentWithAsyncTask extends AsyncTask<String, Void, String> {
            @Override
            protected String doInBackground(String... params) {
                if(params[0].equals("POST")){
                    return sendWtihPost(inputTv.getText().toString());
                }else{
                    return sendWtihGet(inputTv.getText().toString());
                }
            }
            @Override
            protected void onPostExecute(String result) {
                super.onPostExecute(result);
                resultTv.setText(result);
            }
        }
}
```

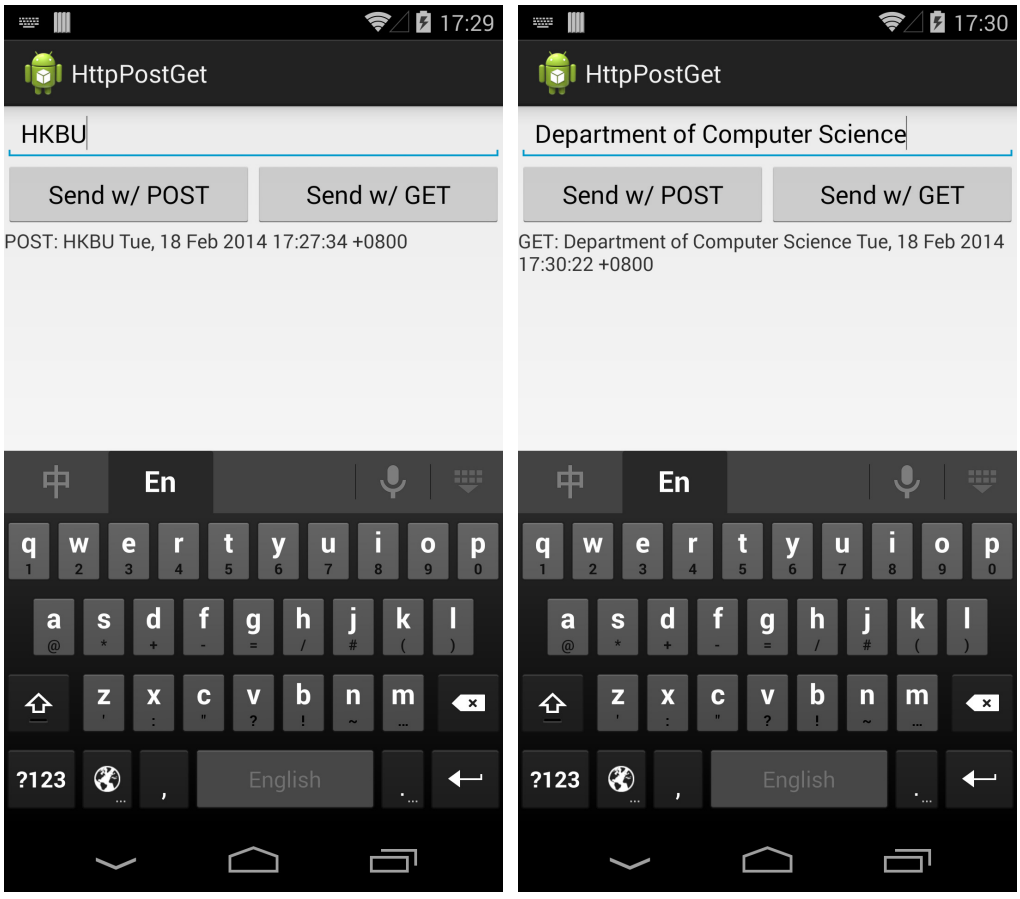
Code explanation:

1. We create two methods **sendWithPost()** and **sendWithGet()** for sending HTTP request using POST and GET respectively.
2. For **sendWithPost(),** we construct a **HttpPost** object with the target URL.
3. We create a **NameValuePair** object and put our name-value pair to an **ArrayList**.
4. We construct an **UrlEncodedFormEntity** with the **NameValuePair ArrayList** and use UTF-8 as an encoding scheme.

5. We set the **UrlEncodedFormEntity** to the **HttpPost** object using **setEntity()**.
6. We create a **DeafultHttpClient** object to execute the **HttpPost** request.
7. We create a **HttpResponse** object to receive the response from **DefaultHttpClient.**
8. When the response code equals to **HTTP 200**, we will get the result from **HttpResponse** object by using **getEntity()** method.
9. We convert the **HttpEntity** object by using the **toString()** in **EntityUtils** imported from **org.apache.http.util.EntityUtils.**
10. For **sendWtihGet(),** the flow is similar to **sendWithPost()** but this time we add the input directly to the URL as an query string.
11. We use **AsyncTask** to execute either **sendWithPost()** or **sendWithGet()** to avoid running network operation in the main thread.
12. We add two **OnClickListeners** for the two buttons respectively in order to trigger the **AsyncTask**.

5. Remember to add **<uses-permission android:name="android.permission.INTERNET"/>** to AndroidManifest.xml

6. Run the app try to input something, press the buttons to get the result below.



7. Source code of **handler.php**:

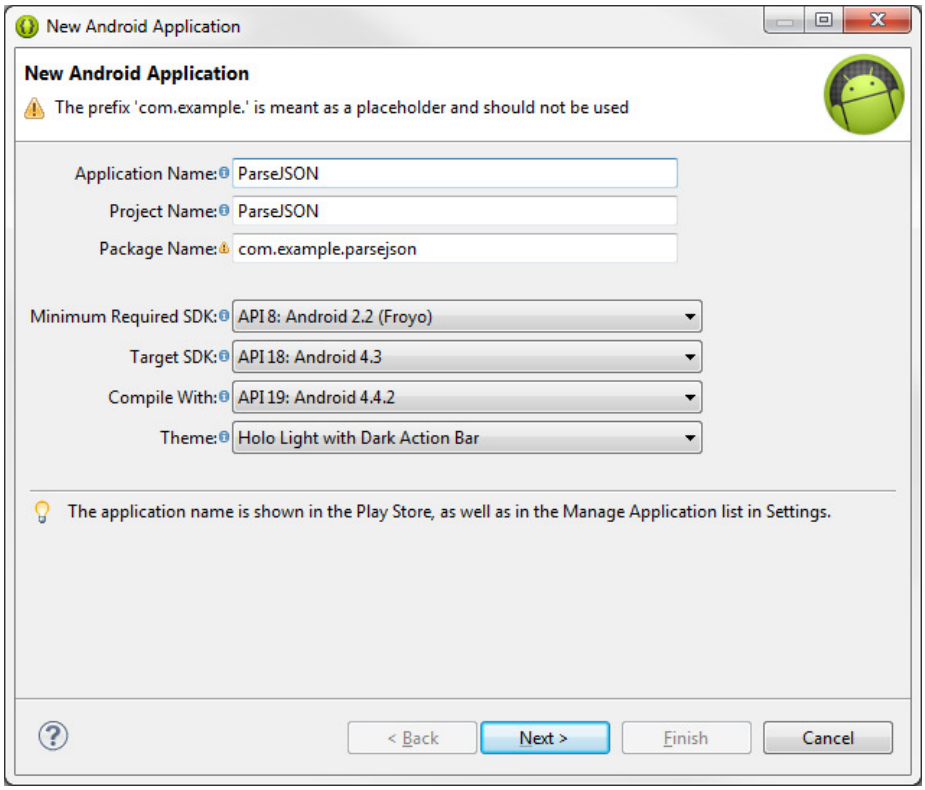
```
<?php
    if(isset($_POST['post_string'])){
        echo "POST: " . $_POST['post_string'] . " " . date("r");
    }else if($_GET['get_string']){
        echo "GET: " . $_GET['get_string']    . " " . date("r");
    }else{
        echo "No data" . " " . date("r");
    }
?>
```

2.2 Retrieve data by parsing JSON

The previous example is only feasible if the data structure is simple. If the data structure we retrieved is complex, it is recommended to wrap it in JSON format. In this example, we will use JSON parser to parse our JSON array located at http://www.comp.hkbu.edu.hk/~sigis/android/public/books.php. Below is the output of our **books.php**:

```
[
  {"isbn":"1234","name":"Book 1","author":"Author 1",
  "publisher":"Publisher 1"},
  {"isbn":"4567","name":"Book 2","author":"Author 2",
  "publisher":"Publisher 2"},
  {"isbn":"7890","name":"Book 3","author":"Author 3",
  "publisher":"Publisher 3"}
]
```

1. Create a new project called **ParseJSON** with **Create activity** option checked in **Configure Project** step**.** Click next in the rest of steps and refer to the screenshot below for the information at the first step.



2. Replace the content in **activity_main.xml** with the following markups:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <Button
        android:id="@+id/parseJSON"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Parse JSON" />

    <TextView
        android:id="@+id/result"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

3. Replace the contents in **MainActivity.java** with the following codes:

```
package com.example.parsejson;
import org.apache.http.HttpResponse;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.util.EntityUtils;
import org.json.JSONArray;
import org.json.JSONObject;
import android.app.Activity;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;

public class MainActivity extends Activity {

 private Button parseJSONBtn;
 private TextView resultTv;

 @Override
 protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_main);
       parseJSONBtn = (Button)findViewById(R.id.parseJSON);
       resultTv = (TextView)findViewById(R.id.result);
       parseJSONBtn.setOnClickListener(parseJSONBtnListener);
 }

 private OnClickListener parseJSONBtnListener    = new OnClickListener(){
       @Override
       public void onClick(View arg0) {
             new GetJSONTask().execute();
       }
 };

 private class GetJSONTask extends AsyncTask<Void, Void, String> {
       @Override
```

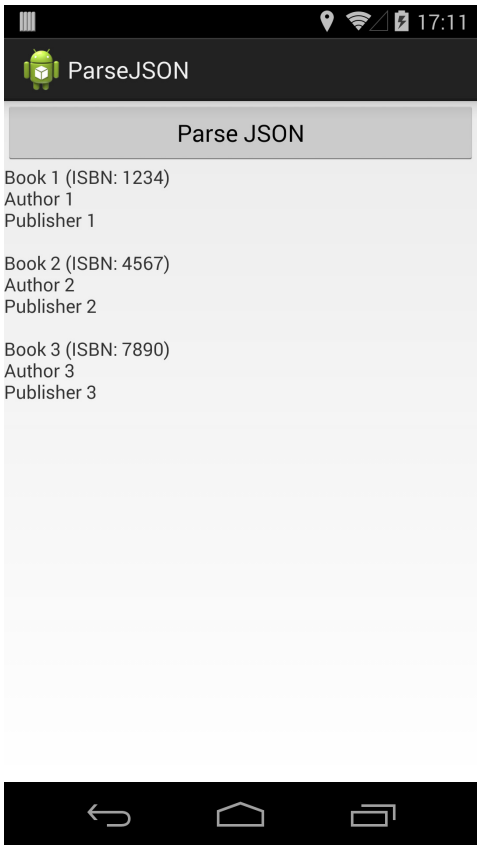
```
        protected String doInBackground(Void... arg) {
            HttpPost request = new HttpPost(

    "http://www.comp.hkbu.edu.hk/~sigis/android/public/books.php");
            String jsonResult = "";
            try {
                HttpResponse httpResponse = new DefaultHttpClient()
                        .execute(request);
                if (httpResponse.getStatusLine().getStatusCode() == 200) {
                    jsonResult = EntityUtils.toString(httpResponse.getEntity());
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
            return jsonResult;
        }

        @Override
        protected void onPostExecute(String jsonResult){
            String str = "";
            try {
                JSONArray jsonArr = new JSONArray(jsonResult);
                for(int i=0; i<jsonArr.length(); i++){
                    JSONObject jsonObj = jsonArr.getJSONObject(i);
                    str += jsonObj.getString("name");
                    str += " (ISBN: " + jsonObj.getString("isbn") + ")\n";
                    str += jsonObj.getString("author") + "\n";
                    str += jsonObj.getString("publisher") + "\n\n";
                }
            }catch(Exception e){
                e.printStackTrace();
            }
            resultTv.setText(str);
        }
     }
    }
```

4. Remember to add **<uses-permission android:name="android.permission.INTERNET"/>** to AndroidManifest.xml

5. After pressed the button, the result should look like:



Code explanation:

1. The output of books.php is a JSON array which contains three JSON objects corresponding to each book.
2. JSON Object is represented using **{ … }**
3. JSON Array is represented using **[…]**
4. We can use **new JSONArray(jsonResult)** to construct a JSON array from a JSON string.
5. We loop through the JSON array and retrieve the JSON objects in each loop.
6. We can use names to retrieve values from each JSON object.

3 Reference and learning resources

- Object oriented programming (OOP) concept
  1. General idea:
     http://en.wikipedia.org/wiki/Object-oriented_programming
  2. OOP in Java
     http://docs.oracle.com/javase/tutorial/java/concepts/

- Java syntax
  - Official Java tutorials
    http://docs.oracle.com/javase/tutorial/java/
  - Java tutorials with web complier
    http://www.learnjavaonline.org/

- Android app development
  - Official Android developer website
    http://developer.android.com/index.html
  - Android development Tutorial (YouTube)
    http://www.youtube.com/watch?v=Z149x12sXsw
  - How to debug Android app in Eclipse (YouTube)
    http://www.youtube.com/watch?v=JqHYbm9e05A

- Seeking development assistance
  - Common questions raised by other developers:
    http://stackoverflow.com/
  - Many Android tutorials with complete source code:
    http://www.anddev.org/

- Introduction of JSON
  http://www.w3schools.com/json/json_intro.asp

- Special Interest Group on Innovative Software homepage:
  http://www.comp.hkbu.edu.hk/~sigis/