We are going see how to make a very simple Android app (in our case, a product inventory app) that will call a PHP script to perform basic CRUD(Create, Read, Update, Delete) operations. To brief you on the architecture, this is how it works. First your android app calls a PHP script in order to perform a data operation, lets say “create”. The PHP script then connects to your MySQL database to perform the operation.  
So the data flows from your Android app to PHP script then finally is stored in your MySQL database. Allright, lets dig deeper.

Please note that the purpose of the code that I have provided here is to, ease you (beginner level) into connecting an Android app with PHP, MYSQL. You should not take this as a standard or secure coding practice. In production environment, you ideally need to avoid any code that will potentially inject vulnerabilities (like MYSQL Injection). MySQL injection itself is a huge topic and cannot be covered in this single post and that is not the agenda of this post either.

**1. What is WAMP Server**

WAMP is acronym for Windows, Apache, MySQL and PHP, Perl, Python. WAMP software is one click installer which creates an environment for developing PHP, MySQL web application. By installing this software you will be installing **Apache**, **MySQL** and **PHP**. Alternatively you can use [XAMP](http://www.apachefriends.org/en/xampp-windows.html) Server also.

### 2. Installing and Running WAMP Server

Download & Install WAMP server from [www.wampserver.com/en/](http://wampserver.com/en/). Once you have installed wamp server, launch the program from Start -> All Programs -> WampServer -> StartWampServer.

You can test your server by opening the address <http://localhost/> in your browser.  
Also you can check phpmyadmin by opening <http://localhost/phpmyadmin>

Following is a screen cast of Downloading and Installing WAMP Server.

**3. Creating and Running PHP Project**

Now you have the environment ready to develop a PHP & MySQL project. Go to the location where you installed WAMP server (In my case i installed in **C:\wamp\**) and go to **www** folder and create a new folder for your project. You have to place all your project files inside this folder.

Create a folder called **android\_connect** and create a new php file called **test.php** and try out simple php code. After placing following code try to open <http://localhost/android_connect/test.php> and you should see a message called “**Welcome, I am connecting Android to PHP, MySQL**“.

**test.php**

|  |
| --- |
| <?php      echo "Welcome, I am connecting Android to PHP, MySQL";  ?> |

Following is a screen cast of Creating and Running a simple PHP project.

**4. Creating MySQL Database and Tables**

In this tutorial i am creating a simple database with one table. Through out this tutorial i am using same table to perform example operations. Now open **phpmyadmin** by opening the address[http://localhost/phpmyadmin/](http://localhost/phpmyadmin) in your browser. You can use the PhpMyAdmin tool to create a database and a table.

I am creating a database named androidhive and a table called products.

|  |
| --- |
| CREATE DATABASE androidhive; |
| CREATE TABLE products(  pid int(11) primary key auto\_increment,  name varchar(100) not null,  price decimal(10,2) not null,  description text,  created\_at timestamp default now(),  updated\_at timestamp  ); |

Following is a screen cast of Creating database and tables in phpmyadmin

**5. Connecting to MySQL database using PHP**

Now the actual server side coding starts. Create a PHP class to connect to MySQL database. The main purpose of this class is to open a connection to database and close the connection whenever its not needed. So create two files called **db\_config.php** and **db\_connect.php**

**db\_config.php** – will have database connection variables  
**db\_connect.php** – a class file to connect to database

Following is code for two php files

**db\_config.php**

|  |
| --- |
| db\_config.php |
| <?php    /\*   \* All database connection variables   \*/    define('DB\_USER', "root"); // db user  define('DB\_PASSWORD', ""); // db password (mention your db password here)  define('DB\_DATABASE', "androidhive"); // database name  define('DB\_SERVER', "localhost"); // db server  ?> |

**db\_connect.php**

|  |
| --- |
| db\_connect.php |
| <?php    /\*\*   \* A class file to connect to database   \*/  class DB\_CONNECT {        // constructor      function \_\_construct() {          // connecting to database          $this->connect();      }        // destructor      function \_\_destruct() {          // closing db connection          $this->close();      }        /\*\*       \* Function to connect with database       \*/      function connect() {          // import database connection variables          require\_once \_\_DIR\_\_ . '/db\_config.php';            // Connecting to mysql database          $con = mysql\_connect(DB\_SERVER, DB\_USER, DB\_PASSWORD) or die(mysql\_error());            // Selecing database          $db = mysql\_select\_db(DB\_DATABASE) or die(mysql\_error()) or die(mysql\_error());            // returing connection cursor          return $con;      }        /\*\*       \* Function to close db connection       \*/      function close() {          // closing db connection          mysql\_close();      }    }    ?> |

**Usage:** When ever you want to connect to MySQL database and do some operations use the db\_connect.php class like this

|  |
| --- |
| $db = new DB\_CONNECT(); // creating class object(will open database connection) |

**6. Basic MySQL CRUD Operations using PHP**

In this tutorial i am covering basic **CRUD (Create, Read, Update, Delete)** operations on MySQL database using PHP.  
If you are a novice about PHP and MySQL i suggest, you to learn basic [PHP](http://www.w3schools.com/php/php_intro.asp) and [SQL](http://www.w3schools.com/sql/sql_syntax.asp) here.

**6.a) Creating a row in MySQL (Creating a new product row)**

In your PHP project create a new php file called **create\_product.php** and place the following code. This file is mainly for creating a new product in products table.

In the following code i am reading product data via POST and storing them in products table. At the end i am echoing appropriate JSON as response.

|  |
| --- |
| create\_product.php |
| <?php    /\*   \* Following code will create a new product row   \* All product details are read from HTTP Post Request   \*/    // array for JSON response  $response = array();    // check for required fields  if (isset($\_POST['name']) && isset($\_POST['price']) && isset($\_POST['description'])) {        $name = $\_POST['name'];      $price = $\_POST['price'];      $description = $\_POST['description'];        // include db connect class      require\_once \_\_DIR\_\_ . '/db\_connect.php';        // connecting to db      $db = new DB\_CONNECT();        // mysql inserting a new row      $result = mysql\_query("INSERT INTO products(name, price, description) VALUES('$name', '$price', '$description')");        // check if row inserted or not      if ($result) {          // successfully inserted into database          $response["success"] = 1;          $response["message"] = "Product successfully created.";            // echoing JSON response          echo json\_encode($response);      } else {          // failed to insert row          $response["success"] = 0;          $response["message"] = "Oops! An error occurred.";            // echoing JSON response          echo json\_encode($response);      }  } else {      // required field is missing      $response["success"] = 0;      $response["message"] = "Required field(s) is missing";        // echoing JSON response      echo json\_encode($response);  }  ?> |

For the above code JSON response will be like

When POST param(s) is missing

|  |
| --- |
| {      "success": 0,      "message": "Required field(s) is missing"  } |

When product is successfully created

|  |
| --- |
| {      "success": 1,      "message": "Product successfully created."  } |

When error occurred while inserting data

|  |
| --- |
| {      "success": 0,      "message": "Oops! An error occurred."  } |

**6.b) Reading a Row from MySQL (Reading product details)**

Create a new php file called get\_product\_details.php and write the following code. This file will get single product details by taking product id (pid) as post parameter.

|  |
| --- |
| get\_product\_details.php |
| <?php    /\*   \* Following code will get single product details   \* A product is identified by product id (pid)   \*/    // array for JSON response  $response = array();    // include db connect class  require\_once \_\_DIR\_\_ . '/db\_connect.php';    // connecting to db  $db = new DB\_CONNECT();    // check for post data  if (isset($\_GET["pid"])) {      $pid = $\_GET['pid'];        // get a product from products table      $result = mysql\_query("SELECT \*FROM products WHERE pid = $pid");        if (!empty($result)) {          // check for empty result          if (mysql\_num\_rows($result) > 0) {                $result = mysql\_fetch\_array($result);                $product = array();              $product["pid"] = $result["pid"];              $product["name"] = $result["name"];              $product["price"] = $result["price"];              $product["description"] = $result["description"];              $product["created\_at"] = $result["created\_at"];              $product["updated\_at"] = $result["updated\_at"];              // success              $response["success"] = 1;                // user node              $response["product"] = array();                array\_push($response["product"], $product);                // echoing JSON response              echo json\_encode($response);          } else {              // no product found              $response["success"] = 0;              $response["message"] = "No product found";                // echo no users JSON              echo json\_encode($response);          }      } else {          // no product found          $response["success"] = 0;          $response["message"] = "No product found";            // echo no users JSON          echo json\_encode($response);      }  } else {      // required field is missing      $response["success"] = 0;      $response["message"] = "Required field(s) is missing";        // echoing JSON response      echo json\_encode($response);  }  ?> |

The json response for the above file will be

When successfully getting product details

|  |
| --- |
| {      "success": 1,      "product": [          {              "pid": "1",              "name": "iPHone 4S",              "price": "300.00",              "description": "iPhone 4S white",              "created\_at": "2012-04-29 01:41:42",              "updated\_at": "0000-00-00 00:00:00"          }      ]  } |

When no product found with matched pid

|  |
| --- |
| {      "success": 0,      "message": "No product found"  } |

**6.c) Reading All Rows from MySQL (Reading all products)**

We need a json to list all the products on android device. So create a new php file named**get\_all\_products.php** and write following code.

|  |
| --- |
| get\_all\_products.php |
| <?php    /\*   \* Following code will list all the products   \*/    // array for JSON response  $response = array();    // include db connect class  require\_once \_\_DIR\_\_ . '/db\_connect.php';    // connecting to db  $db = new DB\_CONNECT();    // get all products from products table  $result = mysql\_query("SELECT \*FROM products") or die(mysql\_error());    // check for empty result  if (mysql\_num\_rows($result) > 0) {      // looping through all results      // products node      $response["products"] = array();        while ($row = mysql\_fetch\_array($result)) {          // temp user array          $product = array();          $product["pid"] = $row["pid"];          $product["name"] = $row["name"];          $product["price"] = $row["price"];          $product["created\_at"] = $row["created\_at"];          $product["updated\_at"] = $row["updated\_at"];            // push single product into final response array          array\_push($response["products"], $product);      }      // success      $response["success"] = 1;        // echoing JSON response      echo json\_encode($response);  } else {      // no products found      $response["success"] = 0;      $response["message"] = "No products found";        // echo no users JSON      echo json\_encode($response);  }  ?> |

And the JSON response for above code

Listing all Products

|  |
| --- |
| {      "products": [          {              "pid": "1",              "name": "iPhone 4S",              "price": "300.00",              "created\_at": "2012-04-29 02:04:02",              "updated\_at": "0000-00-00 00:00:00"          },          {              "pid": "2",              "name": "Macbook Pro",              "price": "600.00",              "created\_at": "2012-04-29 02:04:51",              "updated\_at": "0000-00-00 00:00:00"          },          {              "pid": "3",              "name": "Macbook Air",              "price": "800.00",              "created\_at": "2012-04-29 02:05:57",              "updated\_at": "0000-00-00 00:00:00"          },          {              "pid": "4",              "name": "OS X Lion",              "price": "100.00",              "created\_at": "2012-04-29 02:07:14",              "updated\_at": "0000-00-00 00:00:00"          }      ],      "success": 1  } |

When products not found

|  |
| --- |
| {      "success": 0,      "message": "No products found"  } |

**6.d) Updating a Row in MySQL (Updating product details)**

Create a php file named **update\_product.php** to update product details. Each product is identified by pid.

|  |
| --- |
| update\_product.php |
| <?php    /\*   \* Following code will update a product information   \* A product is identified by product id (pid)   \*/    // array for JSON response  $response = array();    // check for required fields  if (isset($\_POST['pid']) && isset($\_POST['name']) && isset($\_POST['price']) && isset($\_POST['description'])) {        $pid = $\_POST['pid'];      $name = $\_POST['name'];      $price = $\_POST['price'];      $description = $\_POST['description'];        // include db connect class      require\_once \_\_DIR\_\_ . '/db\_connect.php';        // connecting to db      $db = new DB\_CONNECT();        // mysql update row with matched pid      $result = mysql\_query("UPDATE products SET name = '$name', price = '$price', description = '$description' WHERE pid = $pid");        // check if row inserted or not      if ($result) {          // successfully updated          $response["success"] = 1;          $response["message"] = "Product successfully updated.";            // echoing JSON response          echo json\_encode($response);      } else {        }  } else {      // required field is missing      $response["success"] = 0;      $response["message"] = "Required field(s) is missing";        // echoing JSON response      echo json\_encode($response);  }  ?> |

The json reponse of above code, when product is updated successfully

|  |
| --- |
| {      "success": 1,      "message": "Product successfully updated."  } |

**6.e) Deleting a Row in MySQL (Deleting a product)**

The last operation is deletion on database. Create a new php file called delete\_product.php and paste the following code. The main functionality of this file is to delete a product from database.

|  |
| --- |
| delete\_product.php |
| <?php    /\*   \* Following code will delete a product from table   \* A product is identified by product id (pid)   \*/    // array for JSON response  $response = array();    // check for required fields  if (isset($\_POST['pid'])) {      $pid = $\_POST['pid'];        // include db connect class      require\_once \_\_DIR\_\_ . '/db\_connect.php';        // connecting to db      $db = new DB\_CONNECT();        // mysql update row with matched pid      $result = mysql\_query("DELETE FROM products WHERE pid = $pid");        // check if row deleted or not      if (mysql\_affected\_rows() > 0) {          // successfully updated          $response["success"] = 1;          $response["message"] = "Product successfully deleted";            // echoing JSON response          echo json\_encode($response);      } else {          // no product found          $response["success"] = 0;          $response["message"] = "No product found";            // echo no users JSON          echo json\_encode($response);      }  } else {      // required field is missing      $response["success"] = 0;      $response["message"] = "Required field(s) is missing";        // echoing JSON response      echo json\_encode($response);  }  ?> |

When product successfully deleted

|  |
| --- |
| {      "success": 1,      "message": "Product successfully deleted"  } |

When product not found

|  |
| --- |
| {      "success": 0,      "message": "No product found"  } |

Until now, we built a simple api for our products table. We are now done with the server side coding (PHP) and its time to take a break and start our actual android application coding.

**7. Creating Android Application**

Create a new project in your Eclipse IDE by filling the required details.

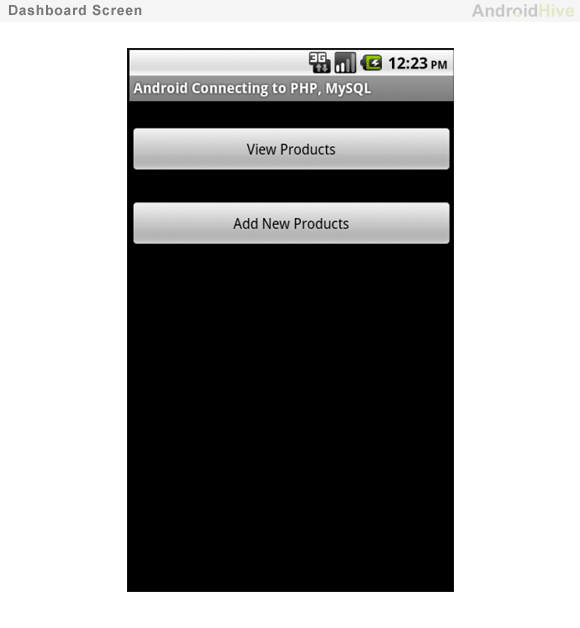
**1**. Create new project in Eclipse IDE by going to **File ⇒ New ⇒ Android Project** and name the Activity class name as **MainScreenActivity**.

**2**. Open your **AndroidManifest.xml** file and add following code. First i am adding all the classes i am creating to manifest file. Also i am adding INTERNET Connect permission.

|  |
| --- |
| AndroidManifest.xml |
| <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="<http://schemas.android.com/apk/res/android>"      package="com.example.androidhive"      android:versionCode="1"      android:versionName="1.0" >        <uses-sdk android:minSdkVersion="8" />        <application          android:configChanges="keyboardHidden|orientation"          android:icon="@drawable/ic\_launcher"          android:label="@string/app\_name" >            <activity              android:name=".MainScreenActivity"              android:label="@string/app\_name" >              <intent-filter>                  <action android:name="android.intent.action.MAIN" />                    <category android:name="android.intent.category.LAUNCHER" />              </intent-filter>          </activity>            <!-- All Product Activity -->          <activity              android:name=".AllProductsActivity"              android:label="All Products" >          </activity>            <!-- Add Product Activity -->          <activity              android:name=".NewProductActivity"              android:label="Add New Product" >          </activity>            <!-- Edit Product Activity -->          <activity              android:name=".EditProductActivity"              android:label="Edit Product" >          </activity>      </application>        <!--  Internet Permissions -->      <uses-permission android:name="android.permission.INTERNET" />    </manifest> |

**3**. Now create a new xml file under **res ⇒ layout** folder and name it as **main\_screen.xml** This layout file contains two simple buttons to view all products and add a new product.

|  |
| --- |
| main\_screen.xml |
| <?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"      android:gravity="center\_horizontal">        <!--  Sample Dashboard screen with Two buttons -->      <!--  Button to view all products screen -->      <Button android:id="@+id/btnViewProducts"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:text="View Products"          android:layout\_marginTop="25dip"/>        <!--  Button to create a new product screen -->      <Button android:id="@+id/btnCreateProduct"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:text="Add New Products"          android:layout\_marginTop="25dip"/>    </LinearLayout> |



**4**. Open you main activity class which is **MainScreenActivity.java** and write click events for two button which are mentioned in **main\_screen.xml** layout.

|  |
| --- |
| MainScreenActivity.java |
| package com.example.androidhive;    import android.app.Activity;  import android.content.Intent;  import android.os.Bundle;  import android.view.View;  import android.widget.Button;    public class MainScreenActivity extends Activity{        Button btnViewProducts;      Button btnNewProduct;        @Override      public void onCreate(Bundle savedInstanceState) {          super.onCreate(savedInstanceState);          setContentView(R.layout.main\_screen);            // Buttons          btnViewProducts = (Button) findViewById(R.id.btnViewProducts);          btnNewProduct = (Button) findViewById(R.id.btnCreateProduct);            // view products click event          btnViewProducts.setOnClickListener(new View.OnClickListener() {                @Override              public void onClick(View view) {                  // Launching All products Activity                  Intent i = new Intent(getApplicationContext(), AllProductsActivity.class);                  startActivity(i);                }          });            // view products click event          btnNewProduct.setOnClickListener(new View.OnClickListener() {                @Override              public void onClick(View view) {                  // Launching create new product activity                  Intent i = new Intent(getApplicationContext(), NewProductActivity.class);                  startActivity(i);                }          });      }  } |

**Displaying All Products in ListView (Read)**

**5**. Now we need an Activity display all the products in list view format. As we know list view needs two xml files, one for listview and other is for single list row. Create two xml files under **res ⇒ layout** folder and name it as **all\_products.xml** and **list\_item.xml**

**all\_products.xml**

|  |
| --- |
| all\_products.xml |
| <?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">      <!-- Main ListView           Always give id value as list(@android:id/list)      -->      <ListView          android:id="@android:id/list"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"/>    </LinearLayout> |

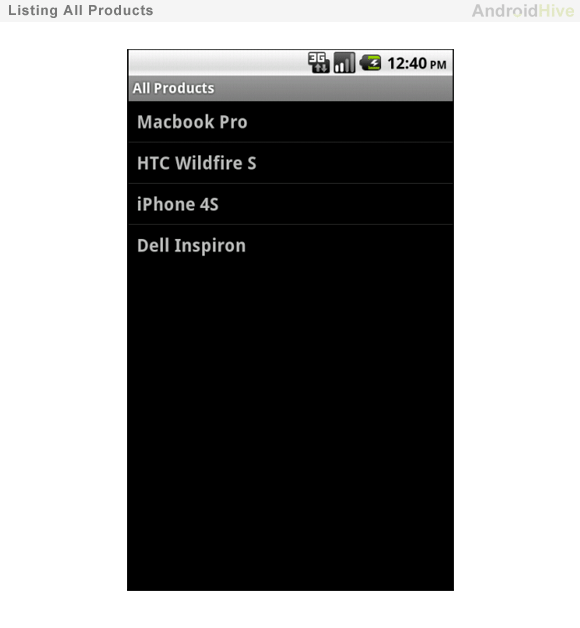
**list\_item.xml**

|  |
| --- |
| list\_item.xml |
| <?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="wrap\_content"      android:orientation="vertical" >        <!-- Product id (pid) - will be HIDDEN - used to pass to other activity -->      <TextView          android:id="@+id/pid"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:visibility="gone" />        <!-- Name Label -->      <TextView          android:id="@+id/name"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:paddingTop="6dip"          android:paddingLeft="6dip"          android:textSize="17dip"          android:textStyle="bold" />    </LinearLayout> |

**6**. Create a new class file and name it as **AllProductsActivity.java**. In the following code

-> First a request is send to **get\_all\_products.php** file using a Background Async task thread.  
-> After getting JSON from get\_all\_products.php, i parsed it and displayed in a listview.  
-> If there are no products found **AddNewProductAcivity** is launched.

|  |
| --- |
| AllProductsActivity.java |
| package com.example.androidhive;    import java.util.ArrayList;  import java.util.HashMap;  import java.util.List;    import org.apache.http.NameValuePair;  import org.json.JSONArray;  import org.json.JSONException;  import org.json.JSONObject;    import android.app.ListActivity;  import android.app.ProgressDialog;  import android.content.Intent;  import android.os.AsyncTask;  import android.os.Bundle;  import android.util.Log;  import android.view.View;  import android.widget.AdapterView;  import android.widget.AdapterView.OnItemClickListener;  import android.widget.ListAdapter;  import android.widget.ListView;  import android.widget.SimpleAdapter;  import android.widget.TextView;    public class AllProductsActivity extends ListActivity {        // Progress Dialog      private ProgressDialog pDialog;        // Creating JSON Parser object      JSONParser jParser = new JSONParser();        ArrayList<HashMap<String, String>> productsList;        // url to get all products list      private static String url\_all\_products = "<http://api.androidhive.info/android_connect/get_all_products.php>";        // JSON Node names      private static final String TAG\_SUCCESS = "success";      private static final String TAG\_PRODUCTS = "products";      private static final String TAG\_PID = "pid";      private static final String TAG\_NAME = "name";        // products JSONArray      JSONArray products = null;        @Override      public void onCreate(Bundle savedInstanceState) {          super.onCreate(savedInstanceState);          setContentView(R.layout.all\_products);            // Hashmap for ListView          productsList = new ArrayList<HashMap<String, String>>();            // Loading products in Background Thread          new LoadAllProducts().execute();            // Get listview          ListView lv = getListView();            // on seleting single product          // launching Edit Product Screen          lv.setOnItemClickListener(new OnItemClickListener() {                @Override              public void onItemClick(AdapterView<?> parent, View view,                      int position, long id) {                  // getting values from selected ListItem                  String pid = ((TextView) view.findViewById(R.id.pid)).getText()                          .toString();                    // Starting new intent                  Intent in = new Intent(getApplicationContext(),                          EditProductActivity.class);                  // sending pid to next activity                  in.putExtra(TAG\_PID, pid);                    // starting new activity and expecting some response back                  startActivityForResult(in, 100);              }          });        }        // Response from Edit Product Activity      @Override      protected void onActivityResult(int requestCode, int resultCode, Intent data) {          super.onActivityResult(requestCode, resultCode, data);          // if result code 100          if (resultCode == 100) {              // if result code 100 is received              // means user edited/deleted product              // reload this screen again              Intent intent = getIntent();              finish();              startActivity(intent);          }        }        /\*\*       \* Background Async Task to Load all product by making HTTP Request       \* \*/      class LoadAllProducts extends AsyncTask<String, String, String> {            /\*\*           \* Before starting background thread Show Progress Dialog           \* \*/          @Override          protected void onPreExecute() {              super.onPreExecute();              pDialog = new ProgressDialog(AllProductsActivity.this);              pDialog.setMessage("Loading products. Please wait...");              pDialog.setIndeterminate(false);              pDialog.setCancelable(false);              pDialog.show();          }            /\*\*           \* getting All products from url           \* \*/          protected String doInBackground(String... args) {              // Building Parameters              List<NameValuePair> params = new ArrayList<NameValuePair>();              // getting JSON string from URL              JSONObject json = jParser.makeHttpRequest(url\_all\_products, "GET", params);                // Check your log cat for JSON reponse              Log.d("All Products: ", json.toString());                try {                  // Checking for SUCCESS TAG                  int success = json.getInt(TAG\_SUCCESS);                    if (success == 1) {                      // products found                      // Getting Array of Products                      products = json.getJSONArray(TAG\_PRODUCTS);                        // looping through All Products                      for (int i = 0; i < products.length(); i++) {                          JSONObject c = products.getJSONObject(i);                            // Storing each json item in variable                          String id = c.getString(TAG\_PID);                          String name = c.getString(TAG\_NAME);                            // creating new HashMap                          HashMap<String, String> map = new HashMap<String, String>();                            // adding each child node to HashMap key => value                          map.put(TAG\_PID, id);                          map.put(TAG\_NAME, name);                            // adding HashList to ArrayList                          productsList.add(map);                      }                  } else {                      // no products found                      // Launch Add New product Activity                      Intent i = new Intent(getApplicationContext(),                              NewProductActivity.class);                      // Closing all previous activities                      i.addFlags(Intent.FLAG\_ACTIVITY\_CLEAR\_TOP);                      startActivity(i);                  }              } catch (JSONException e) {                  e.printStackTrace();              }                return null;          }            /\*\*           \* After completing background task Dismiss the progress dialog           \* \*\*/          protected void onPostExecute(String file\_url) {              // dismiss the dialog after getting all products              pDialog.dismiss();              // updating UI from Background Thread              runOnUiThread(new Runnable() {                  public void run() {                      /\*\*                       \* Updating parsed JSON data into ListView                       \* \*/                      ListAdapter adapter = new SimpleAdapter(                              AllProductsActivity.this, productsList,                              R.layout.list\_item, new String[] { TAG\_PID,                                      TAG\_NAME},                              new int[] { R.id.pid, R.id.name });                      // updating listview                      setListAdapter(adapter);                  }              });            }        }  } |



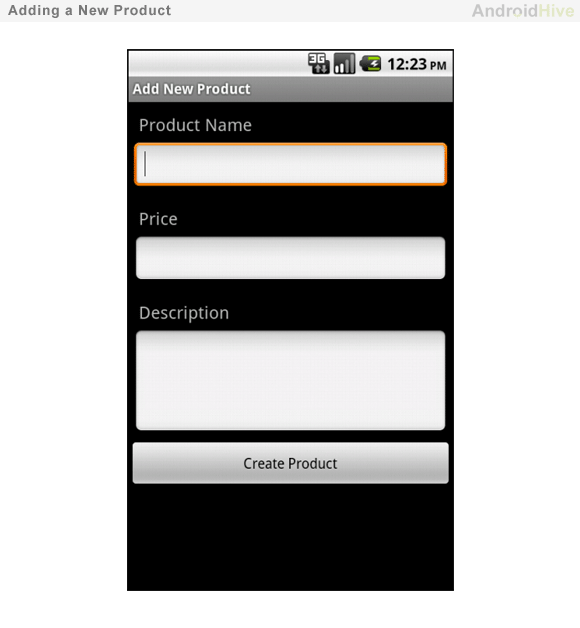
**Adding a New Product (Write)**

**7**. Create a new view and acivity to add a new product into mysql database. Create a simple form which contains EditText for product name, price and description.

Create a new xml file and name it as **add\_product.xml** and paste the following code to create a simple form.

**add\_product.xml**

|  |
| --- |
| add\_product.xml |
| <?xml version="1.0" encoding="utf-8"?>  <LinearLayout xmlns:android="<http://schemas.android.com/apk/res/android>"      android:layout\_width="match\_parent"      android:layout\_height="match\_parent"      android:orientation="vertical" >        <!-- Name Label -->      <TextView android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:text="Product Name"          android:paddingLeft="10dip"          android:paddingRight="10dip"          android:paddingTop="10dip"          android:textSize="17dip"/>        <!-- Input Name -->      <EditText android:id="@+id/inputName"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:layout\_margin="5dip"          android:layout\_marginBottom="15dip"          android:singleLine="true"/>        <!-- Price Label -->      <TextView android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:text="Price"          android:paddingLeft="10dip"          android:paddingRight="10dip"          android:paddingTop="10dip"          android:textSize="17dip"/>        <!-- Input Price -->      <EditText android:id="@+id/inputPrice"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:layout\_margin="5dip"          android:layout\_marginBottom="15dip"          android:singleLine="true"          android:inputType="numberDecimal"/>        <!-- Description Label -->      <TextView android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:text="Description"          android:paddingLeft="10dip"          android:paddingRight="10dip"          android:paddingTop="10dip"          android:textSize="17dip"/>        <!-- Input description -->      <EditText android:id="@+id/inputDesc"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:layout\_margin="5dip"          android:layout\_marginBottom="15dip"          android:lines="4"          android:gravity="top"/>        <!-- Button Create Product -->      <Button android:id="@+id/btnCreateProduct"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:text="Create Product"/>    </LinearLayout> |



**8**. Now create new Activity to insert a new product into mysql database. Create a class file and name it as **NewProductActivity.java** and type the following code. In the following code

-> First new product data is read from the EditText form and formatted into a basic params.  
-> A request is made to **create\_product.php** to create a new product through HTTP post.  
-> After getting json response from create\_product.php, If success bit is 1 then list view is refreshed with newly added product.

|  |
| --- |
| NewProductActivity.java |
| package com.example.androidhive;    import java.util.ArrayList;  import java.util.List;    import org.apache.http.NameValuePair;  import org.apache.http.message.BasicNameValuePair;  import org.json.JSONException;  import org.json.JSONObject;    import android.app.Activity;  import android.app.ProgressDialog;  import android.content.Intent;  import android.os.AsyncTask;  import android.os.Bundle;  import android.util.Log;  import android.view.View;  import android.widget.Button;  import android.widget.EditText;    public class NewProductActivity extends Activity {        // Progress Dialog      private ProgressDialog pDialog;        JSONParser jsonParser = new JSONParser();      EditText inputName;      EditText inputPrice;      EditText inputDesc;        // url to create new product      private static String url\_create\_product = "<http://api.androidhive.info/android_connect/create_product.php>";        // JSON Node names      private static final String TAG\_SUCCESS = "success";        @Override      public void onCreate(Bundle savedInstanceState) {          super.onCreate(savedInstanceState);          setContentView(R.layout.add\_product);            // Edit Text          inputName = (EditText) findViewById(R.id.inputName);          inputPrice = (EditText) findViewById(R.id.inputPrice);          inputDesc = (EditText) findViewById(R.id.inputDesc);            // Create button          Button btnCreateProduct = (Button) findViewById(R.id.btnCreateProduct);            // button click event          btnCreateProduct.setOnClickListener(new View.OnClickListener() {                @Override              public void onClick(View view) {                  // creating new product in background thread                  new CreateNewProduct().execute();              }          });      }        /\*\*       \* Background Async Task to Create new product       \* \*/      class CreateNewProduct extends AsyncTask<String, String, String> {            /\*\*           \* Before starting background thread Show Progress Dialog           \* \*/          @Override          protected void onPreExecute() {              super.onPreExecute();              pDialog = new ProgressDialog(NewProductActivity.this);              pDialog.setMessage("Creating Product..");              pDialog.setIndeterminate(false);              pDialog.setCancelable(true);              pDialog.show();          }            /\*\*           \* Creating product           \* \*/          protected String doInBackground(String... args) {              String name = inputName.getText().toString();              String price = inputPrice.getText().toString();              String description = inputDesc.getText().toString();                // Building Parameters              List<NameValuePair> params = new ArrayList<NameValuePair>();              params.add(new BasicNameValuePair("name", name));              params.add(new BasicNameValuePair("price", price));              params.add(new BasicNameValuePair("description", description));                // getting JSON Object              // Note that create product url accepts POST method              JSONObject json = jsonParser.makeHttpRequest(url\_create\_product,                      "POST", params);                // check log cat fro response              Log.d("Create Response", json.toString());                // check for success tag              try {                  int success = json.getInt(TAG\_SUCCESS);                    if (success == 1) {                      // successfully created product                      Intent i = new Intent(getApplicationContext(), AllProductsActivity.class);                      startActivity(i);                        // closing this screen                      finish();                  } else {                      // failed to create product                  }              } catch (JSONException e) {                  e.printStackTrace();              }                return null;          }            /\*\*           \* After completing background task Dismiss the progress dialog           \* \*\*/          protected void onPostExecute(String file\_url) {              // dismiss the dialog once done              pDialog.dismiss();          }        }  } |

**Reading, Updating and Deleting a Single Product**

**9**. If you notice the AllProductsActivity.java, In listview i am launching EditProductAcivity.java once a single list item is selected. So create xml file called **edit\_product.xml** and create a form which is same as create\_product.xml.

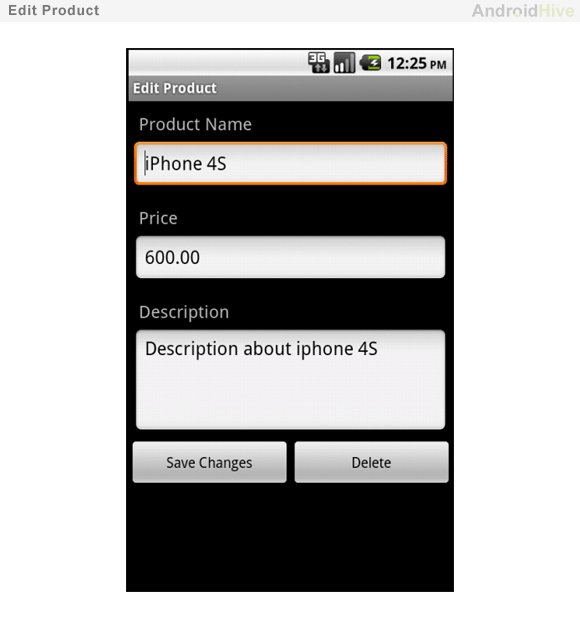
**edit\_product.xml**

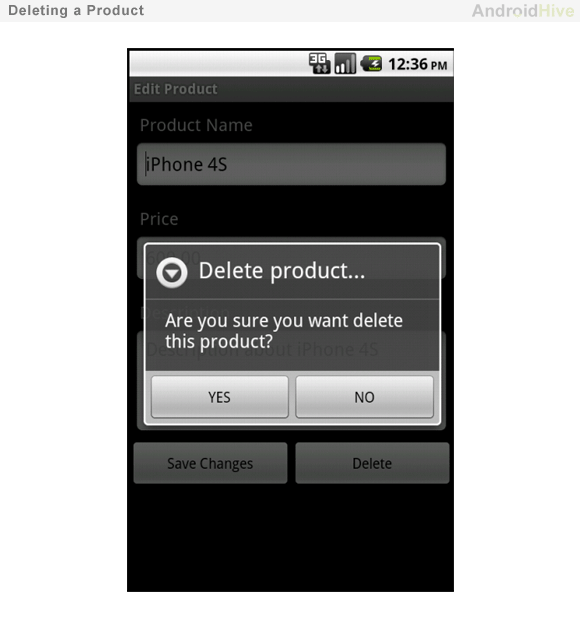
|  |
| --- |
| edit\_product.xml |
| <?xml version="1.0" encoding="utf-8"?>  <LinearLayout xmlns:android="<http://schemas.android.com/apk/res/android>"      android:layout\_width="match\_parent"      android:layout\_height="match\_parent"      android:orientation="vertical" >        <!-- Name Label -->      <TextView android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:text="Product Name"          android:paddingLeft="10dip"          android:paddingRight="10dip"          android:paddingTop="10dip"          android:textSize="17dip"/>        <!-- Input Name -->      <EditText android:id="@+id/inputName"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:layout\_margin="5dip"          android:layout\_marginBottom="15dip"          android:singleLine="true"/>        <!-- Price Label -->      <TextView android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:text="Price"          android:paddingLeft="10dip"          android:paddingRight="10dip"          android:paddingTop="10dip"          android:textSize="17dip"/>        <!-- Input Price -->      <EditText android:id="@+id/inputPrice"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:layout\_margin="5dip"          android:layout\_marginBottom="15dip"          android:singleLine="true"          android:inputType="numberDecimal"/>        <!-- Description Label -->      <TextView android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:text="Description"          android:paddingLeft="10dip"          android:paddingRight="10dip"          android:paddingTop="10dip"          android:textSize="17dip"/>        <!-- Input description -->      <EditText android:id="@+id/inputDesc"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:layout\_margin="5dip"          android:layout\_marginBottom="15dip"          android:lines="4"          android:gravity="top"/>        <LinearLayout android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:orientation="horizontal">          <!-- Button Create Product -->      <Button android:id="@+id/btnSave"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:text="Save Changes"          android:layout\_weight="1"/>        <!-- Button Create Product -->      <Button android:id="@+id/btnDelete"          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:text="Delete"          android:layout\_weight="1"/>      </LinearLayout>    </LinearLayout> |

**10**. Create a class file for edit\_product.xml and name it as **EditProductActivity.java** and fill it with following code. In the following code

-> First product id (**pid**) is read from the intent which is sent from listview.  
-> A request is made to **get\_product\_details.php** and after getting product details in json format, I parsed the json and displayed in EditText.  
-> After displaying product data in the form if user clicks on Save Changes Button, another HTTP request is made to **update\_product.php** to store updated product data.  
-> If the user selected Delete Product Button, HTTP request is made to **delete\_product.php** and product is deleted from mysql database, and listview is refreshed with new product list.

|  |
| --- |
| EditProductActivity.java |
| package com.example.androidhive;    import java.util.ArrayList;  import java.util.List;    import org.apache.http.NameValuePair;  import org.apache.http.message.BasicNameValuePair;  import org.json.JSONArray;  import org.json.JSONException;  import org.json.JSONObject;    import android.app.Activity;  import android.app.ProgressDialog;  import android.content.Intent;  import android.os.AsyncTask;  import android.os.Bundle;  import android.util.Log;  import android.view.View;  import android.widget.Button;  import android.widget.EditText;    public class EditProductActivity extends Activity {        EditText txtName;      EditText txtPrice;      EditText txtDesc;      EditText txtCreatedAt;      Button btnSave;      Button btnDelete;        String pid;        // Progress Dialog      private ProgressDialog pDialog;        // JSON parser class      JSONParser jsonParser = new JSONParser();        // single product url      private static final String url\_product\_detials = "<http://api.androidhive.info/android_connect/get_product_details.php>";        // url to update product      private static final String url\_update\_product = "<http://api.androidhive.info/android_connect/update_product.php>";        // url to delete product      private static final String url\_delete\_product = "<http://api.androidhive.info/android_connect/delete_product.php>";        // JSON Node names      private static final String TAG\_SUCCESS = "success";      private static final String TAG\_PRODUCT = "product";      private static final String TAG\_PID = "pid";      private static final String TAG\_NAME = "name";      private static final String TAG\_PRICE = "price";      private static final String TAG\_DESCRIPTION = "description";        @Override      public void onCreate(Bundle savedInstanceState) {          super.onCreate(savedInstanceState);          setContentView(R.layout.edit\_product);            // save button          btnSave = (Button) findViewById(R.id.btnSave);          btnDelete = (Button) findViewById(R.id.btnDelete);            // getting product details from intent          Intent i = getIntent();            // getting product id (pid) from intent          pid = i.getStringExtra(TAG\_PID);            // Getting complete product details in background thread          new GetProductDetails().execute();            // save button click event          btnSave.setOnClickListener(new View.OnClickListener() {                @Override              public void onClick(View arg0) {                  // starting background task to update product                  new SaveProductDetails().execute();              }          });            // Delete button click event          btnDelete.setOnClickListener(new View.OnClickListener() {                @Override              public void onClick(View arg0) {                  // deleting product in background thread                  new DeleteProduct().execute();              }          });        }        /\*\*       \* Background Async Task to Get complete product details       \* \*/      class GetProductDetails extends AsyncTask<String, String, String> {            /\*\*           \* Before starting background thread Show Progress Dialog           \* \*/          @Override          protected void onPreExecute() {              super.onPreExecute();              pDialog = new ProgressDialog(EditProductActivity.this);              pDialog.setMessage("Loading product details. Please wait...");              pDialog.setIndeterminate(false);              pDialog.setCancelable(true);              pDialog.show();          }            /\*\*           \* Getting product details in background thread           \* \*/          protected String doInBackground(String... params) {                // updating UI from Background Thread              runOnUiThread(new Runnable() {                  public void run() {                      // Check for success tag                      int success;                      try {                          // Building Parameters                          List<NameValuePair> params = new ArrayList<NameValuePair>();                          params.add(new BasicNameValuePair("pid", pid));                            // getting product details by making HTTP request                          // Note that product details url will use GET request                          JSONObject json = jsonParser.makeHttpRequest(                                  url\_product\_detials, "GET", params);                            // check your log for json response                          Log.d("Single Product Details", json.toString());                            // json success tag                          success = json.getInt(TAG\_SUCCESS);                          if (success == 1) {                              // successfully received product details                              JSONArray productObj = json                                      .getJSONArray(TAG\_PRODUCT); // JSON Array                                // get first product object from JSON Array                              JSONObject product = productObj.getJSONObject(0);                                // product with this pid found                              // Edit Text                              txtName = (EditText) findViewById(R.id.inputName);                              txtPrice = (EditText) findViewById(R.id.inputPrice);                              txtDesc = (EditText) findViewById(R.id.inputDesc);                                // display product data in EditText                              txtName.setText(product.getString(TAG\_NAME));                              txtPrice.setText(product.getString(TAG\_PRICE));                              txtDesc.setText(product.getString(TAG\_DESCRIPTION));                            }else{                              // product with pid not found                          }                      } catch (JSONException e) {                          e.printStackTrace();                      }                  }              });                return null;          }            /\*\*           \* After completing background task Dismiss the progress dialog           \* \*\*/          protected void onPostExecute(String file\_url) {              // dismiss the dialog once got all details              pDialog.dismiss();          }      }        /\*\*       \* Background Async Task to  Save product Details       \* \*/      class SaveProductDetails extends AsyncTask<String, String, String> {            /\*\*           \* Before starting background thread Show Progress Dialog           \* \*/          @Override          protected void onPreExecute() {              super.onPreExecute();              pDialog = new ProgressDialog(EditProductActivity.this);              pDialog.setMessage("Saving product ...");              pDialog.setIndeterminate(false);              pDialog.setCancelable(true);              pDialog.show();          }            /\*\*           \* Saving product           \* \*/          protected String doInBackground(String... args) {                // getting updated data from EditTexts              String name = txtName.getText().toString();              String price = txtPrice.getText().toString();              String description = txtDesc.getText().toString();                // Building Parameters              List<NameValuePair> params = new ArrayList<NameValuePair>();              params.add(new BasicNameValuePair(TAG\_PID, pid));              params.add(new BasicNameValuePair(TAG\_NAME, name));              params.add(new BasicNameValuePair(TAG\_PRICE, price));              params.add(new BasicNameValuePair(TAG\_DESCRIPTION, description));                // sending modified data through http request              // Notice that update product url accepts POST method              JSONObject json = jsonParser.makeHttpRequest(url\_update\_product,                      "POST", params);                // check json success tag              try {                  int success = json.getInt(TAG\_SUCCESS);                    if (success == 1) {                      // successfully updated                      Intent i = getIntent();                      // send result code 100 to notify about product update                      setResult(100, i);                      finish();                  } else {                      // failed to update product                  }              } catch (JSONException e) {                  e.printStackTrace();              }                return null;          }            /\*\*           \* After completing background task Dismiss the progress dialog           \* \*\*/          protected void onPostExecute(String file\_url) {              // dismiss the dialog once product uupdated              pDialog.dismiss();          }      }        /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*       \* Background Async Task to Delete Product       \* \*/      class DeleteProduct extends AsyncTask<String, String, String> {            /\*\*           \* Before starting background thread Show Progress Dialog           \* \*/          @Override          protected void onPreExecute() {              super.onPreExecute();              pDialog = new ProgressDialog(EditProductActivity.this);              pDialog.setMessage("Deleting Product...");              pDialog.setIndeterminate(false);              pDialog.setCancelable(true);              pDialog.show();          }            /\*\*           \* Deleting product           \* \*/          protected String doInBackground(String... args) {                // Check for success tag              int success;              try {                  // Building Parameters                  List<NameValuePair> params = new ArrayList<NameValuePair>();                  params.add(new BasicNameValuePair("pid", pid));                    // getting product details by making HTTP request                  JSONObject json = jsonParser.makeHttpRequest(                          url\_delete\_product, "POST", params);                    // check your log for json response                  Log.d("Delete Product", json.toString());                    // json success tag                  success = json.getInt(TAG\_SUCCESS);                  if (success == 1) {                      // product successfully deleted                      // notify previous activity by sending code 100                      Intent i = getIntent();                      // send result code 100 to notify about product deletion                      setResult(100, i);                      finish();                  }              } catch (JSONException e) {                  e.printStackTrace();              }                return null;          }            /\*\*           \* After completing background task Dismiss the progress dialog           \* \*\*/          protected void onPostExecute(String file\_url) {              // dismiss the dialog once product deleted              pDialog.dismiss();            }        }  } |





**JSON Parser Class**

I used a JSON Parser class to get JSON from URL. This class supports two http request methods GET and POST to get json from url.

|  |
| --- |
| JSONParser.java |
| package com.example.androidhive;    import java.io.BufferedReader;  import java.io.IOException;  import java.io.InputStream;  import java.io.InputStreamReader;  import java.io.UnsupportedEncodingException;  import java.util.List;    import org.apache.http.HttpEntity;  import org.apache.http.HttpResponse;  import org.apache.http.NameValuePair;  import org.apache.http.client.ClientProtocolException;  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.client.utils.URLEncodedUtils;  import org.apache.http.impl.client.DefaultHttpClient;  import org.json.JSONException;  import org.json.JSONObject;    import android.util.Log;    public class JSONParser {        static InputStream is = null;      static JSONObject jObj = null;      static String json = "";        // constructor      public JSONParser() {        }        // function get json from url      // by making HTTP POST or GET mehtod      public JSONObject makeHttpRequest(String url, String method,              List<NameValuePair> params) {            // Making HTTP request          try {                // check for request method              if(method == "POST"){                  // request method is POST                  // defaultHttpClient                  DefaultHttpClient httpClient = new DefaultHttpClient();                  HttpPost httpPost = new HttpPost(url);                  httpPost.setEntity(new UrlEncodedFormEntity(params));                    HttpResponse httpResponse = httpClient.execute(httpPost);                  HttpEntity httpEntity = httpResponse.getEntity();                  is = httpEntity.getContent();                }else if(method == "GET"){                  // request method is GET                  DefaultHttpClient httpClient = new DefaultHttpClient();                  String paramString = URLEncodedUtils.format(params, "utf-8");                  url += "?" + paramString;                  HttpGet httpGet = new HttpGet(url);                    HttpResponse httpResponse = httpClient.execute(httpGet);                  HttpEntity httpEntity = httpResponse.getEntity();                  is = httpEntity.getContent();              }            } catch (UnsupportedEncodingException e) {              e.printStackTrace();          } catch (ClientProtocolException e) {              e.printStackTrace();          } catch (IOException e) {              e.printStackTrace();          }            try {              BufferedReader reader = new BufferedReader(new InputStreamReader(                      is, "iso-8859-1"), 8);              StringBuilder sb = new StringBuilder();              String line = null;              while ((line = reader.readLine()) != null) {                  sb.append(line + "\n");              }              is.close();              json = sb.toString();          } catch (Exception e) {              Log.e("Buffer Error", "Error converting result " + e.toString());          }            // try parse the string to a JSON object          try {              jObj = new JSONObject(json);          } catch (JSONException e) {              Log.e("JSON Parser", "Error parsing data " + e.toString());          }            // return JSON String          return jObj;        }  } |

Run your project and test the application. You might get lot of errors. Always use **Log Cat** to debug your application, and if you couldn’t solve your errors please do comment here.