**//to check whether network is connected or not.**

**public static boolean** isNetworkConnected(Context ctx) {  
 **if** (ctx == **null**) {  
 **return false**;  
 }  
 ConnectivityManager cm = (ConnectivityManager) ctx.getApplicationContext()  
 .getSystemService(Context.***CONNECTIVITY\_SERVICE***);  
  
 NetworkInfo wifiNetwork = cm.getNetworkInfo(ConnectivityManager.***TYPE\_WIFI***);  
 **if** (wifiNetwork != **null** && wifiNetwork.isConnected()) {  
 **return true**;  
 }  
  
 NetworkInfo mobileNetwork = cm.getNetworkInfo(ConnectivityManager.***TYPE\_MOBILE***);  
 **if** (mobileNetwork != **null** && mobileNetwork.isConnected()) {  
 **return true**;  
 }  
  
 NetworkInfo activeNetwork = cm.getActiveNetworkInfo();  
 **if** (activeNetwork != **null** && activeNetwork.isConnected()) {  
 **return true**;  
 }  
 **return false**;  
}

// to check the system permission

**if** (!Utils.*checkPermission*(**ctx**)) {  
 Utils.*requestPermission*(MainActivity.**this**);  
 } **else** {  
}

**private static final int *PERMISSION\_REQUEST\_CODE*** = 200;

**public static boolean** checkPermission(Context ctx) {  
 **int** result = ContextCompat.*checkSelfPermission*(ctx, ***ACCESS\_FINE\_LOCATION***);  
 **int** result1 = ContextCompat.*checkSelfPermission*(ctx, ***CAMERA***);  
 **int** result2 = ContextCompat.*checkSelfPermission*(ctx, ***WRITE\_EXTERNAL\_STORAGE***);  
 **int** result3 = ContextCompat.*checkSelfPermission*(ctx, ***READ\_CONTACTS***);  
 **int** result4 = ContextCompat.*checkSelfPermission*(ctx, ***RECEIVE\_SMS***);  
 **int** result5 = ContextCompat.*checkSelfPermission*(ctx, ***READ\_EXTERNAL\_STORAGE***);  
 **return** result == PackageManager.***PERMISSION\_GRANTED*** && result1 == PackageManager.***PERMISSION\_GRANTED*** && result2 == PackageManager.***PERMISSION\_GRANTED*** && result3 == PackageManager.***PERMISSION\_GRANTED*** && result4 == PackageManager.***PERMISSION\_GRANTED*** && result5 == PackageManager.***PERMISSION\_GRANTED***;  
}

**public static void** requestPermission(Activity activity) {  
 ActivityCompat.*requestPermissions*(activity,  
 **new** String[]{***ACCESS\_FINE\_LOCATION***, ***CAMERA***, ***WRITE\_EXTERNAL\_STORAGE***, ***READ\_CONTACTS***, ***RECEIVE\_SMS***, ***READ\_EXTERNAL\_STORAGE***}, ***PERMISSION\_REQUEST\_CODE***);  
}

//device information

**String device\_unique\_id =** Build.***SERIAL***;

**int gcm\_key** = Build.VERSION.***SDK\_INT***; *// API Level*

String **device\_make** = Build.***MANUFACTURER***;*// Company*

String device\_**device\_model**= Build.***MODEL***;*// Device Model*

String **device\_os\_version**= Build.VERSION.***RELEASE***;*// OS*

**device\_type**= "android"

**app\_version** ="0.1"

//location enabler

**public static void** mLocationEnabler(Context ctx) {  
 LocationManager lm = (LocationManager) ctx.getSystemService(Context.***LOCATION\_SERVICE***);  
 **boolean** gps\_enabled = **false**;  
 **boolean** network\_enabled = **false**;  
 **try** {  
 gps\_enabled = lm.isProviderEnabled(LocationManager.***GPS\_PROVIDER***);  
 } **catch** (Exception ex) {  
 }  
 **try** {  
 network\_enabled = lm.isProviderEnabled(LocationManager.***NETWORK\_PROVIDER***);  
 } **catch** (Exception ex) {  
 }  
 **if** (!gps\_enabled && !network\_enabled) {  
 LocationEnableDialog dlg = **new** LocationEnableDialog(ctx);  
 dlg.show();  
 dlg.setCancelable(**false**);  
*// dlg.setCanceledOnTouchOutside(false);* }  
 }

**tv\_enable**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent myIntent = **new** Intent(Settings.***ACTION\_LOCATION\_SOURCE\_SETTINGS***);  
 **ctx**.startActivity(myIntent);  
 dismiss();  
 }  
});

//latlong to address

**public void** mAddressofpresent(Double lat, Double longii)  
 **throws** IOException {  
 Geocoder geocoder = **new** Geocoder(**ctx**, Locale.*getDefault*());  
 List<Address> addresses = geocoder.getFromLocation(lat, longii, 1);  
 **if** (addresses.size() != 0) {  
 String address = addresses.get(0).getAddressLine(0);  
 String city = addresses.get(0).getAddressLine(1);  
 **tv\_presentlocation**.setText(address + **" "** + city);  
 } **else** {  
  
 }  
}

//service call using jsonobject

String json = **""**;

JSONObject jsonObject = **new** JSONObject();  
**try** {  
 jsonObject.accumulate(**"UserID"**, **user\_id**);  
 jsonObject.accumulate(**"token"**, **token**);  
 jsonObject.accumulate(**"category"**, category);  
  
 json = jsonObject.toString();  
  
 String data = **"{"** + **" \"data\" "** + **":"** + json + **"}"**;

**if** (Utils.*isNetworkConnected*(**ctx**)) {  
 Asyntask asynce = **new** Asyntask(**ctx**, **new** URL(Constants.*base\_url* + Constants.*friendslist\_url*), **fraglistener**);  
 asynce.execute(data);  
 } **else** {  
 Utils.*mCustomToast*(**ctx**, **"Check Your Internet Connectivity.."**, 1);  
 }  
  
} **catch** (JSONException e1) {  
 e1.printStackTrace();  
} **catch** (MalformedURLException e) {  
 e.printStackTrace();  
}