**import** java.io.IOException;  
**import** java.io.InputStreamReader;  
**import** java.net.HttpURLConnection;  
**import** java.net.MalformedURLException;  
**import** java.net.URL;  
**import** java.net.URLEncoder;  
**import** java.util.ArrayList;  
**import** java.lang.String;  
  
**import** org.json.JSONArray;  
**import** org.json.JSONException;  
**import** org.json.JSONObject;  
  
**import** android.app.Activity;  
**import** android.content.Context;  
**import** android.os.Bundle;  
**import** android.util.Log;  
**import** android.view.View;  
**import** android.widget.AdapterView;  
**import** android.widget.AdapterView.OnItemClickListener;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.AutoCompleteTextView;  
**import** android.widget.Filter;  
**import** android.widget.Filterable;  
**import** android.widget.Toast;  
  
**import** com.google.android.gms.common.api.GoogleApiClient;  
**import** com.example.tonia.projecttraveldiaries.R;  
  
**public class** TravelDiaries **extends** Activity **implements** OnItemClickListener {  
  
 **private static final** String ***LOG\_TAG*** = **"GooglePlacesAutocomplet"**;  
 **private static final** String ***PLACES\_API\_BASE*** = **"https://maps.googleapis.com/maps/api/place"**;  
 **private static final** String ***TYPE\_AUTOCOMPLETE*** = **"/autocomplete"**;  
 **private static final** String ***OUT\_JSON*** = **"/json"**;  
  
 **private static final** String ***API\_KEY*** = **"AIzaSyAEZFfAbiNaBz191THLeDWQQDUm3\_ry55w"**;  
  
 @Override  
 **public void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 AutoCompleteTextView autoCompView = (AutoCompleteTextView) findViewById(R.id.***autoCompleteTextView***);  
  
 autoCompView.setAdapter(**new** GooglePlacesAutocompleteAdapter(**this**, R.layout.***list\_item***));  
 autoCompView.setOnItemClickListener(**this**);  
 }  
  
 **public void** onItemClick(AdapterView adapterView, View view, **int** position, **long** id) {  
 String str = (String) adapterView.getItemAtPosition(position);  
 Toast.*makeText*(**this**, str, Toast.***LENGTH\_SHORT***).show();  
 }  
  
 **public static** ArrayList autocomplete(String input) {  
 ArrayList resultList = **null**;  
  
 HttpURLConnection conn = **null**;  
 StringBuilder jsonResults = **new** StringBuilder();  
 **try** {  
 StringBuilder sb = **new** StringBuilder(***PLACES\_API\_BASE*** + ***TYPE\_AUTOCOMPLETE*** + ***OUT\_JSON***);  
 sb.append(**"?key="** + ***API\_KEY***);  
 System.***out***.println(**"Begin"**);  
 sb.append(**"&components=country:us"**);  
 sb.append(**"&input="** + URLEncoder.*encode*(input, **"utf8"**));  
  
 URL url = **new** URL(sb.toString());  
 System.***out***.println(**"After url"**);  
 conn = (HttpURLConnection) url.openConnection();  
 System.***out***.println(**"After connection"**);  
 InputStreamReader in = **new** InputStreamReader(conn.getInputStream());  
 System.***out***.println(**"After stream reader"**);  
  
 *// Load the results into a StringBuilder* **int** read;  
 **char**[] buff = **new char**[1024];  
 **while** ((read = in.read(buff)) != -1) {  
 System.***out***.println(**"Hi"**);  
 jsonResults.append(buff, 0, read);  
 }  
 } **catch** (MalformedURLException e) {  
 Log.*e*(***LOG\_TAG***, **"Error processing Places API URL"**, e);  
 **return** resultList;  
 } **catch** (IOException e) {  
 Log.*e*(***LOG\_TAG***, **"Error connecting to Places API"**, e);  
 **return** resultList;  
 } **finally** {  
 System.***out***.println(**"Finally-1"**);  
 **if** (conn != **null**) {  
 conn.disconnect();  
 System.***out***.println(**"Finally-2"**);  
 }  
 System.***out***.println(**"Finally-3"**);  
 }  
 System.***out***.println(**"Atleast here"**);  
 **try** {  
 System.***out***.println(**"XXXX"**);  
 *// Create a JSON object hierarchy from the results* JSONObject jsonObj = **new** JSONObject(jsonResults.toString());  
 JSONArray predsJsonArray = jsonObj.getJSONArray(**"predictions"**);  
 System.***out***.println(**"lenth is "**+predsJsonArray.length());  
   
 *// Extract the Place descriptions from the results* resultList = **new** ArrayList(predsJsonArray.length());  
 **for** (**int** i = 0; i < predsJsonArray.length(); i++) {  
 System.***out***.println(predsJsonArray.getJSONObject(i).getString(**"description"**));  
 System.***out***.println(**"============================================================"**);  
 resultList.add(predsJsonArray.getJSONObject(i).getString(**"description"**));  
 }  
 } **catch** (JSONException e) {  
 Log.*e*(***LOG\_TAG***, **"Cannot process JSON results"**, e);  
 }  
 System.***out***.println(**"Finally here"**);  
 **return** resultList;  
 }  
  
 **class** GooglePlacesAutocompleteAdapter **extends** ArrayAdapter **implements** Filterable {  
 **private** ArrayList **resultList**;  
  
 **public** GooglePlacesAutocompleteAdapter(Context context, **int** textViewResourceId) {  
 **super**(context, textViewResourceId);  
 }  
  
 @Override  
 **public int** getCount() {  
 **return resultList**.size();  
 }  
  
 @Override  
 **public** String getItem(**int** index) {  
 **return** (String) **resultList**.get(index);  
 }  
  
 @Override  
 **public** Filter getFilter() {  
 Filter filter = **new** Filter() {  
 @Override  
 **protected** FilterResults performFiltering(CharSequence constraint) {  
 FilterResults filterResults = **new** FilterResults();  
 **if** (constraint != **null**) {  
 *// Retrieve the autocomplete results.* **resultList** = *autocomplete*(constraint.toString());  
  
 *// Assign the data to the FilterResults* filterResults.**values** = **resultList**;  
 filterResults.**count** = **resultList**.size();  
 }  
 **return** filterResults;  
 }  
  
 @Override  
 **protected void** publishResults(CharSequence constraint, FilterResults results) {  
 **if** (results != **null** && results.**count** > 0) {  
 notifyDataSetChanged();  
 } **else** {  
 notifyDataSetInvalidated();  
 }  
 }  
 };  
 **return** filter;  
 }  
 }  
}

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical" android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**>  
  
 <**AutoCompleteTextView  
 android:id="@+id/autoCompleteTextView"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="@string/From"**/>  
  
  
  
</**LinearLayout**>

Strings.xml

<**resources**>  
 <**string name="app\_name"**>Travel Diaries</**string**>  
  
 <**string name="hello\_world"**>Hello world!</**string**>  
 <**string name="action\_settings"**>Settings</**string**>  
 <**string name="From"**>Enter a place</**string**>  
 <**string name="To"**>Enter destination</**string**>  
 <**string name="Chk\_Restaurant"**>Restaurants</**string**>  
 <**string name="Chk\_Bars"**>Bars</**string**>  
 <**string name="Chk\_TouristPlaces"**>TouristPlaces</**string**>

</resources>

<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools" android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent" android:paddingLeft="@dimen/activity\_horizontal\_margin"  
 android:id="@+id/Background"  
 android:orientation="horizontal"  
 android:paddingRight="@dimen/activity\_horizontal\_margin"  
 android:paddingTop="@dimen/activity\_vertical\_margin"  
 android:paddingBottom="@dimen/activity\_vertical\_margin" tools:context=".LayoutExample"**>  
  
 <**AutoCompleteTextView  
 android:id="@+id/Source"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="@string/From"**/>  
  
 <**AutoCompleteTextView  
 android:id="@+id/Destination"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="@string/To"** />  
  
 <**Button  
 android:id="@+id/Search"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="match\_parent"  
 android:text="button1"** />  
  
 <**CheckBox  
 android:id="@+id/Restaurants"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/Chk\_Restaurant"**/>  
  
 <**CheckBox  
 android:id="@+id/Bars"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/Chk\_Bars"**/>  
  
 <**CheckBox  
 android:id="@+id/Tourist"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/Chk\_TouristPlaces"**/>  
  
</**LinearLayout**>

**package** com.example.tonia.projecttraveldiaries;  
  
*/\*\*  
 \* Created by Tonia on 5/14/15.  
 \*/***import** java.util.ArrayList;  
**import** java.util.HashMap;  
**import** java.util.List;  
  
**import** org.json.JSONArray;  
**import** org.json.JSONException;  
**import** org.json.JSONObject;  
  
**public class** PlaceJSONParser {  
  
 */\*\* Receives a JSONObject and returns a list \*/* **public** List<HashMap<String,String>> parse(JSONObject jObject){  
  
 JSONArray jPlaces = **null**;  
 **try** {  
 */\*\* Retrieves all the elements in the 'places' array \*/* jPlaces = jObject.getJSONArray(**"results"**);  
 } **catch** (JSONException e) {  
 e.printStackTrace();  
 }  
 */\*\* Invoking getPlaces with the array of json object  
 \* where each json object represent a place  
 \*/* **return** getPlaces(jPlaces);  
 }  
  
 **private** List<HashMap<String, String>> getPlaces(JSONArray jPlaces){  
 **int** placesCount = jPlaces.length();  
 List<HashMap<String, String>> placesList = **new** ArrayList<HashMap<String,String>>();  
 HashMap<String, String> place = **null**;  
  
 */\*\* Taking each place, parses and adds to list object \*/* **for**(**int** i=0; i<placesCount;i++){  
 **try** {  
 */\*\* Call getPlace with place JSON object to parse the place \*/* place = getPlace((JSONObject)jPlaces.get(i));  
 placesList.add(place);  
  
 } **catch** (JSONException e) {  
 e.printStackTrace();  
 }  
 }  
  
 **return** placesList;  
 }  
  
 */\*\* Parsing the Place JSON object \*/* **private** HashMap<String, String> getPlace(JSONObject jPlace){  
  
 HashMap<String, String> place = **new** HashMap<String, String>();  
 String placeName = **"-NA-"**;  
 String vicinity=**"-NA-"**;  
 String latitude=**""**;  
 String longitude=**""**;  
  
 **try** {  
 *// Extracting Place name, if available* **if**(!jPlace.isNull(**"name"**)){  
 placeName = jPlace.getString(**"name"**);  
 }  
  
 *// Extracting Place Vicinity, if available* **if**(!jPlace.isNull(**"vicinity"**)){  
 vicinity = jPlace.getString(**"vicinity"**);  
 }  
  
 latitude = jPlace.getJSONObject(**"geometry"**).getJSONObject(**"location"**).getString(**"lat"**);  
 longitude = jPlace.getJSONObject(**"geometry"**).getJSONObject(**"location"**).getString(**"lng"**);  
  
 place.put(**"place\_name"**, placeName);  
 place.put(**"vicinity"**, vicinity);  
 place.put(**"lat"**, latitude);  
 place.put(**"lng"**, longitude);  
  
 } **catch** (JSONException e) {  
 e.printStackTrace();  
 }  
 **return** place;  
 }  
}

**package** com.example.tonia.projecttraveldiaries;  
  
**import** java.io.BufferedReader;  
**import** java.io.IOException;  
**import** java.io.InputStream;  
**import** java.io.InputStreamReader;  
**import** java.net.HttpURLConnection;  
**import** java.net.URL;  
**import** java.util.HashMap;  
**import** java.util.List;  
  
**import** org.json.JSONObject;  
  
**import** android.app.Dialog;  
**import** android.location.Criteria;  
**import** android.location.Location;  
**import** android.location.LocationListener;  
**import** android.location.LocationManager;  
**import** android.os.AsyncTask;  
**import** android.os.Bundle;  
**import** android.support.v4.app.FragmentActivity;  
**import** android.util.Log;  
**import** android.view.Menu;  
**import** android.view.View;  
**import** android.view.View.OnClickListener;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.Button;  
**import** android.widget.Spinner;  
  
**import** com.google.android.gms.common.ConnectionResult;  
**import** com.google.android.gms.common.GooglePlayServicesUtil;  
**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.LatLng;  
**import** com.google.android.gms.maps.model.MarkerOptions;  
**import** com.example.tonia.projecttraveldiaries.R;  
  
**public class** TravelDiaries **extends** FragmentActivity **implements** LocationListener{  
  
 GoogleMap **mGoogleMap**;  
 Spinner **mSprPlaceType**;  
  
 String[] **mPlaceType**=**null**;  
 String[] **mPlaceTypeName**=**null**;  
  
 **double mLatitude**=0;  
 **double mLongitude**=0;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
 *// Array of place types* **mPlaceType** = getResources().getStringArray(R.array.***place\_type***);  
  
 *// Array of place type names* **mPlaceTypeName** = getResources().getStringArray(R.array.***place\_type\_name***);  
  
 *// Creating an array adapter with an array of Place types  
 // to populate the spinner* ArrayAdapter<String> adapter = **new** ArrayAdapter<String>(**this**, android.R.layout.***simple\_spinner\_dropdown\_item***, **mPlaceTypeName**);  
  
 *// Getting reference to the Spinner* **mSprPlaceType** = (Spinner) findViewById(R.id.***spr\_place\_type***);  
  
 *// Setting adapter on Spinner to set place types* **mSprPlaceType**.setAdapter(adapter);  
  
 Button btnFind;  
  
 *// Getting reference to Find Button* btnFind = ( Button ) findViewById(R.id.***btn\_find***);  
  
 *// Getting Google Play availability status* **int** status = GooglePlayServicesUtil.*isGooglePlayServicesAvailable*(getBaseContext());  
  
 **if**(status!=ConnectionResult.***SUCCESS***){ *// Google Play Services are not available* **int** requestCode = 10;  
 Dialog dialog = GooglePlayServicesUtil.*getErrorDialog*(status, **this**, requestCode);  
 dialog.show();  
  
 }**else** { *// Google Play Services are available  
  
 // Getting reference to the SupportMapFragment* SupportMapFragment fragment = ( SupportMapFragment) getSupportFragmentManager().findFragmentById(R.id.***map***);  
  
 *// Getting Google Map* **mGoogleMap** = fragment.getMap();  
  
 *// Enabling MyLocation in Google Map* **mGoogleMap**.setMyLocationEnabled(**true**);  
  
 *// Getting LocationManager object from System Service LOCATION\_SERVICE* LocationManager locationManager = (LocationManager) getSystemService(***LOCATION\_SERVICE***);  
  
 *// Creating a criteria object to retrieve provider* Criteria criteria = **new** Criteria();  
  
 *// Getting the name of the best provider* String provider = locationManager.getBestProvider(criteria, **true**);  
  
 *// Getting Current Location From GPS* Location location = locationManager.getLastKnownLocation(provider);  
  
 **if**(location!=**null**){  
 onLocationChanged(location);  
 }  
  
 locationManager.requestLocationUpdates(provider, 20000, 0, **this**);  
  
 *// Setting click event lister for the find button* btnFind.setOnClickListener(**new** OnClickListener() {  
  
 @Override  
 **public void** onClick(View v) {  
  
 **int** selectedPosition = **mSprPlaceType**.getSelectedItemPosition();  
 String type = **mPlaceType**[selectedPosition];  
 **final** String YOUR\_API\_KEY = **"AIzaSyA5ekVcroQmV9SuUsuX37RO74lHeLg4CoE"**;  
  
 StringBuilder sb = **new** StringBuilder(**"https://maps.googleapis.com/maps/api/place/nearbysearch/json?"**);  
 sb.append(**"location="**+**mLatitude**+**","**+**mLongitude**);  
 sb.append(**"&radius=5000"**);  
 sb.append(**"&types="**+type);  
 sb.append(**"&sensor=true"**);  
 sb.append(**"&key="**+YOUR\_API\_KEY);  
  
 *// Creating a new non-ui thread task to download json data* PlacesTask placesTask = **new** PlacesTask();  
  
 *// Invokes the "doInBackground()" method of the class PlaceTask* placesTask.execute(sb.toString());  
  
 }  
 });  
  
 }  
  
 }  
  
 */\*\* A method to download json data from url \*/* **private** String downloadUrl(String strUrl) **throws** IOException{  
 String data = **""**;  
 InputStream iStream = **null**;  
 HttpURLConnection urlConnection = **null**;  
 **try**{  
 URL url = **new** URL(strUrl);  
  
 *// Creating an http connection to communicate with url* urlConnection = (HttpURLConnection) url.openConnection();  
  
 *// Connecting to url* urlConnection.connect();  
  
 *// Reading data from url* iStream = urlConnection.getInputStream();  
  
 BufferedReader br = **new** BufferedReader(**new** InputStreamReader(iStream));  
  
 StringBuffer sb = **new** StringBuffer();  
  
 String line = **""**;  
 **while**( ( line = br.readLine()) != **null**){  
 sb.append(line);  
 }  
  
 data = sb.toString();  
  
 br.close();  
  
 }**catch**(Exception e){  
 Log.*d*(**"Exception downloading"**, e.toString());  
 }**finally**{  
 iStream.close();  
 urlConnection.disconnect();  
 }  
  
 **return** data;  
 }  
  
 */\*\* A class, to download Google Places \*/* **private class** PlacesTask **extends** AsyncTask<String, Integer, String>{  
  
 String **data** = **null**;  
  
 *// Invoked by execute() method of this object* @Override  
 **protected** String doInBackground(String... url) {  
 **try**{  
 **data** = downloadUrl(url[0]);  
 }**catch**(Exception e){  
 Log.*d*(**"Background Task"**,e.toString());  
 }  
 **return data**;  
 }  
  
 *// Executed after the complete execution of doInBackground() method* @Override  
 **protected void** onPostExecute(String result){  
 ParserTask parserTask = **new** ParserTask();  
  
 *// Start parsing the Google places in JSON format  
 // Invokes the "doInBackground()" method of the class ParseTask* parserTask.execute(result);  
 }  
  
 }  
  
 */\*\* A class to parse the Google Places in JSON format \*/* **private class** ParserTask **extends** AsyncTask<String, Integer, List<HashMap<String,String>>>{  
  
 JSONObject **jObject**;  
  
 *// Invoked by execute() method of this object* @Override  
 **protected** List<HashMap<String,String>> doInBackground(String... jsonData) {  
  
 List<HashMap<String, String>> places = **null**;  
 PlaceJSONParser placeJsonParser = **new** PlaceJSONParser();  
  
 **try**{  
 **jObject** = **new** JSONObject(jsonData[0]);  
  
 */\*\* Getting the parsed data as a List construct \*/* places = placeJsonParser.parse(**jObject**);  
  
 }**catch**(Exception e){  
 Log.*d*(**"Exception"**,e.toString());  
 }  
 **return** places;  
 }  
  
 *// Executed after the complete execution of doInBackground() method* @Override  
 **protected void** onPostExecute(List<HashMap<String,String>> list){  
  
 *// Clears all the existing markers* **mGoogleMap**.clear();  
  
 **for**(**int** i=0;i<list.size();i++){  
  
 *// Creating a marker* MarkerOptions markerOptions = **new** MarkerOptions();  
  
 *// Getting a place from the places list* HashMap<String, String> hmPlace = list.get(i);  
  
 *// Getting latitude of the place* **double** lat = Double.*parseDouble*(hmPlace.get(**"lat"**));  
  
 *// Getting longitude of the place* **double** lng = Double.*parseDouble*(hmPlace.get(**"lng"**));  
  
 *// Getting name* String name = hmPlace.get(**"place\_name"**);  
  
 *// Getting vicinity* String vicinity = hmPlace.get(**"vicinity"**);  
  
 LatLng latLng = **new** LatLng(lat, lng);  
  
 *// Setting the position for the marker* markerOptions.position(latLng);  
  
 *// Setting the title for the marker.  
 //This will be displayed on taping the marker* markerOptions.title(name + **" : "** + vicinity);  
  
 *// Placing a marker on the touched position* **mGoogleMap**.addMarker(markerOptions);  
 }  
 }  
 }  
  
 @Override  
 **public void** onLocationChanged(Location location) {  
 **mLatitude** = location.getLatitude();  
 **mLongitude** = location.getLongitude();  
 LatLng latLng = **new** LatLng(**mLatitude**, **mLongitude**);  
  
 **mGoogleMap**.moveCamera(CameraUpdateFactory.*newLatLng*(latLng));  
 **mGoogleMap**.animateCamera(CameraUpdateFactory.*zoomTo*(12));  
 }  
  
 @Override  
 **public void** onProviderDisabled(String provider) {  
 *//* ***TODO Auto-generated method stub*** }  
  
 @Override  
 **public void** onProviderEnabled(String provider) {  
 *//* ***TODO Auto-generated method stub*** }  
  
 @Override  
 **public void** onStatusChanged(String provider, **int** status, Bundle extras) {  
 *//* ***TODO Auto-generated method stub*** }  
}