Open Data App

Development of an Android Open Data App based on the actual available repository from the City of Malaga.

Araceli Manzano Chicano

aramanzano@uma.es

Abstract. The views about the world digital future can be expressed with one word: data. The government open data repositories constitute an important source of updated information that serves as a basis for a big variety of Smart City applications. In this text, a simple Android application which uses environmental data from Malaga City Hall open data repository will be developed.

1. Introduction

According to the *Open definition,* a statement of principles around open information, open data is *"data that can be freely used, modified and shared by anyone for any purpose"* [1]. Many different entities, such as government, private sector or academia, can open their data in several forms including financial data, geographic data, environmental data or even archives and collections metadata. After data is opened by any entity, it can be freely reused by others to create new knowledge and new applications of information for commercial and non-commercial use.

In this text, data collected and published by Spanish Government at municipal level, more concretely at the Malaga City Hall Open Data repository which can be found in [2]. While the infrastructure around the publication of open data can be considered robust, just a few mechanisms have already been established to ascertain the innovative use of government open data [3]. This innovative use can be materialized in products, services, processes, management, market and social interventions and behaviour.

In what concerns sustainability, government – released open data is fuelling a whole new level of innovative Smart City applications that promote a change in the social behaviour towards more respectful actuations with environment. One of the fundamental issues in this field is the recycling of waste from urban and industrial activity.

Recycling is the process of taking used material and processing it to make something new or reconditioning it. This is considered as a very environmentally friendly process since in addition to avoiding the waste of potentially useful materials, it reduces the consumption of fresh raw materials and energy usage. Pollution levels are limited because waste material is not incinerated or buried. Many kinds of materials can be recycled, such as glass, paper, metal, plastics, textiles and electronics, but many highly developed countries still have a very poor record with respect to recycling [4].

For this use case, a simple Android application based on environmental open data is designed and implemented with the main objective of demonstrate the use and visualization of government open data for Smart City applications, specifically in the field of sustainability.

2. App Design and Implementation

The design of the *"¿Dónde reciclo?"* App consist in a main activity with the double role of presentation screen and menu screen. The rest of the application activities are called from this main activity and its number could be modified attending to the number of available datasets in the connected repository.

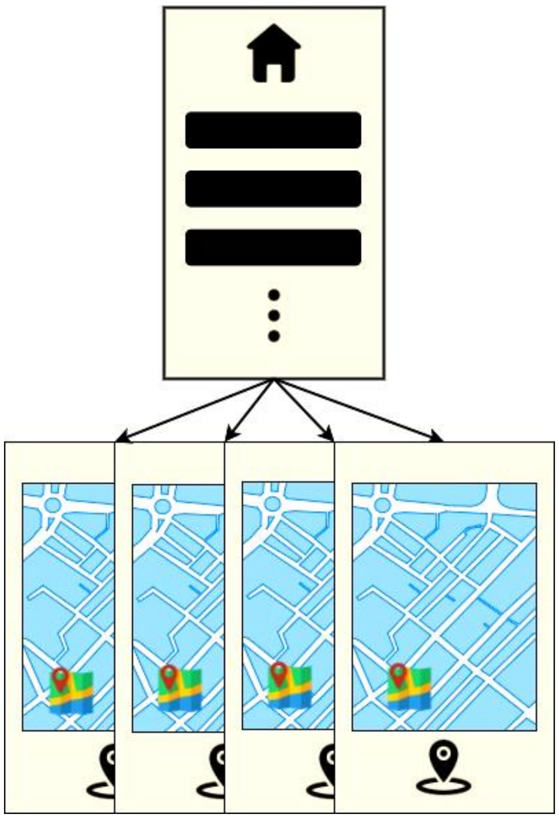


Figure 1: Basic scheme of the Android Open Data App.

In this first approach, since only the datasets for solid urban waste, plastics, paper/carton and industrial waste are available in the Malaga City Hall open data repository, the app main activity, shown in Figure 2, presents four buttons to access the activity corresponding to each available dataset. In addition, it has another button with the Limasa company's logo in order to make the end user’s access to further information about recycling and waste treatment in the City of Malaga easier, as it is shown in Figure 3.

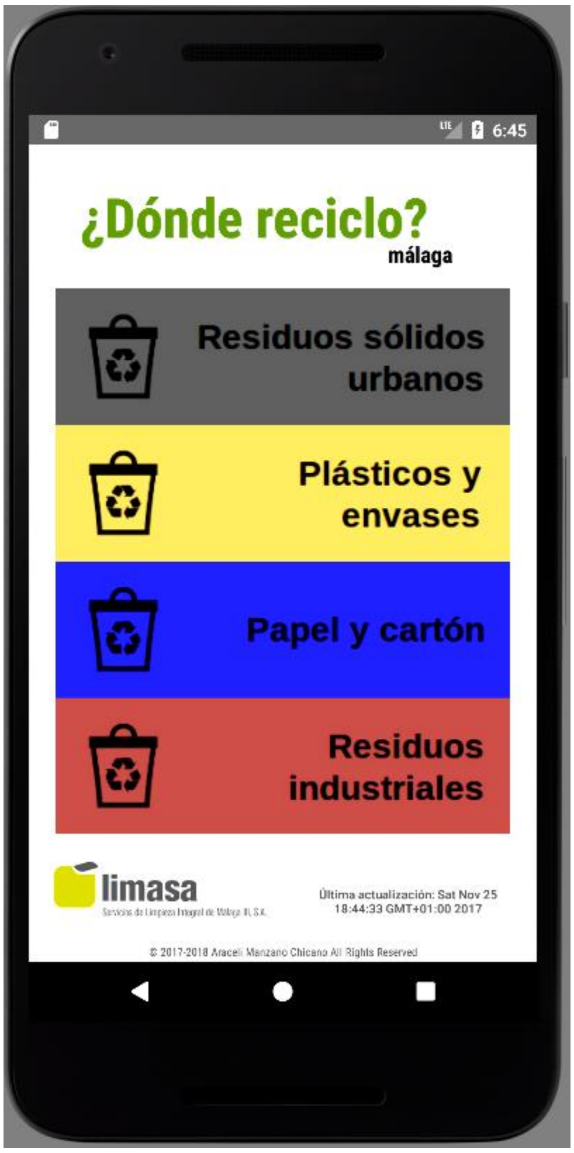

6:45
¿Dónde reciclo?
málaga
Residuos sólidos urbanos
Plásticos y envases
Papel y cartón
Residuos industriales
limasa
Servicios de Limpieza Integral de Málaga III, S.A.
Última actualización: Sat Nov 25 18:44:33 GMT+01:00 2017
© 2017-2018 Araceli Manzano Chicano All Rights Reserved

Figure 2: Main Activity.

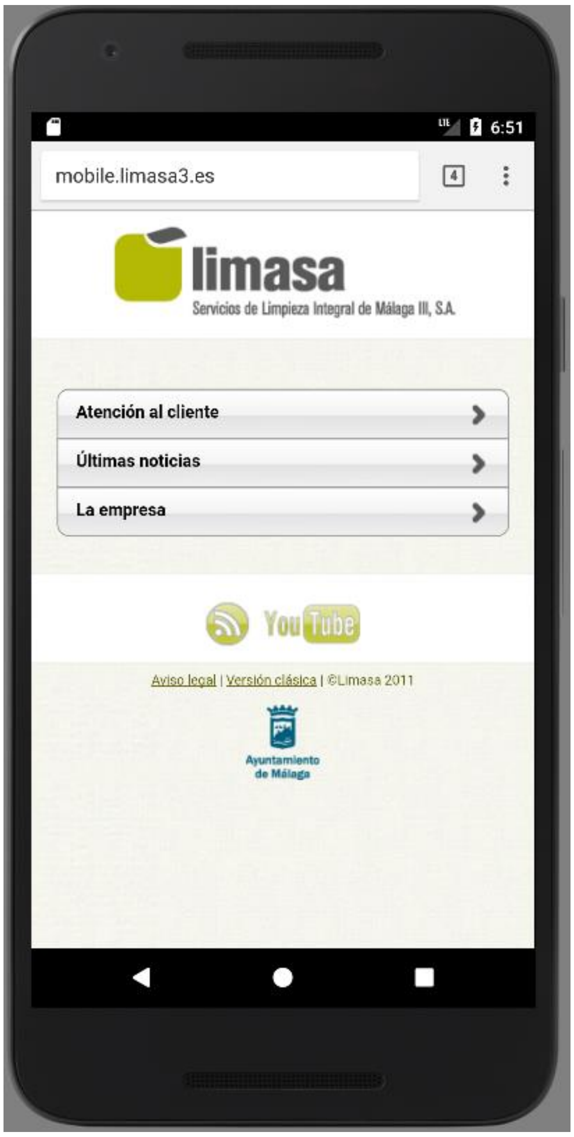


Figure 3: Direct Link to Limasa webpage.

Once the end user access any of the buttons associated to the type of waste to know its recycle points locations, a different secondary activity emerges with a basic scheme formed by a visualization of all the recycle points locations on a Google Map fragment and an option to find the nearby recycle point related to the end user current location. For this purpose, user permissions to access the device location are asked when the activity emerge. Figures 4 and 5 shown the different activities associated to each type of waste to be recycled and their corresponding visualizations.

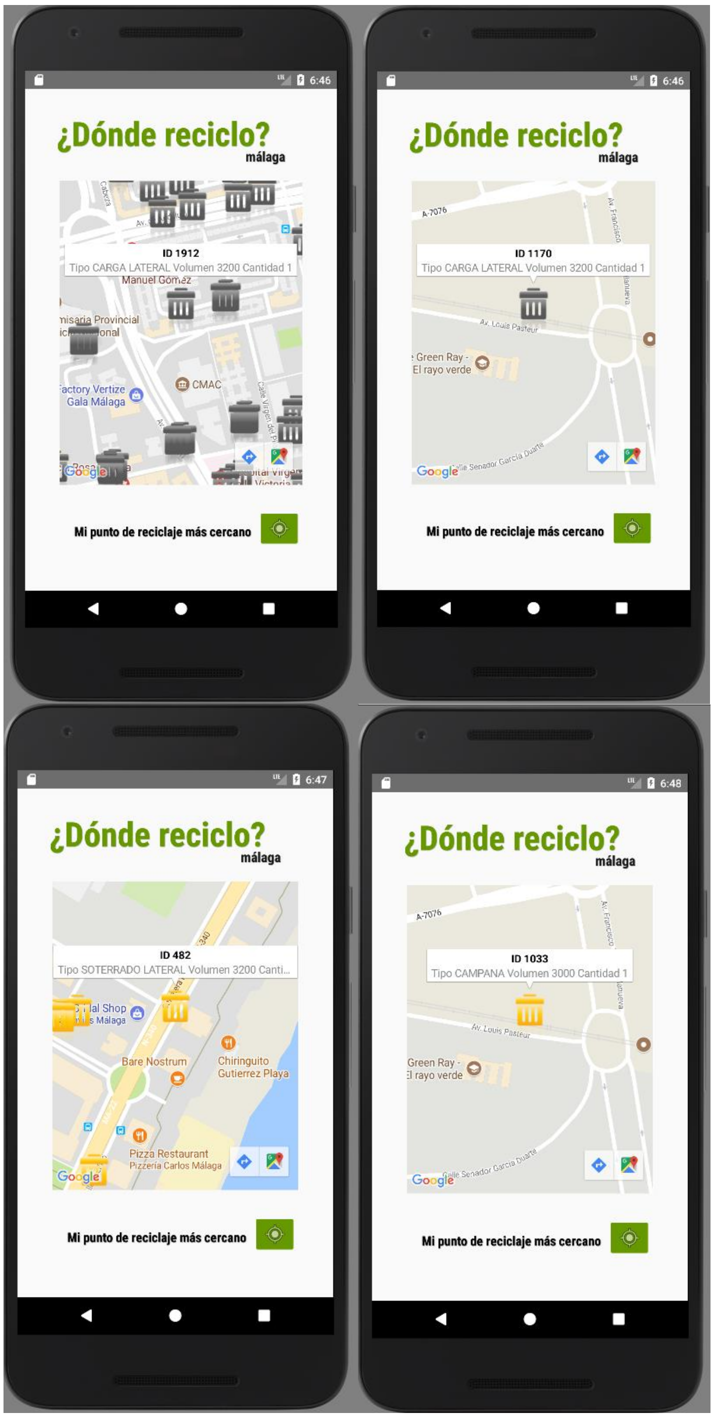
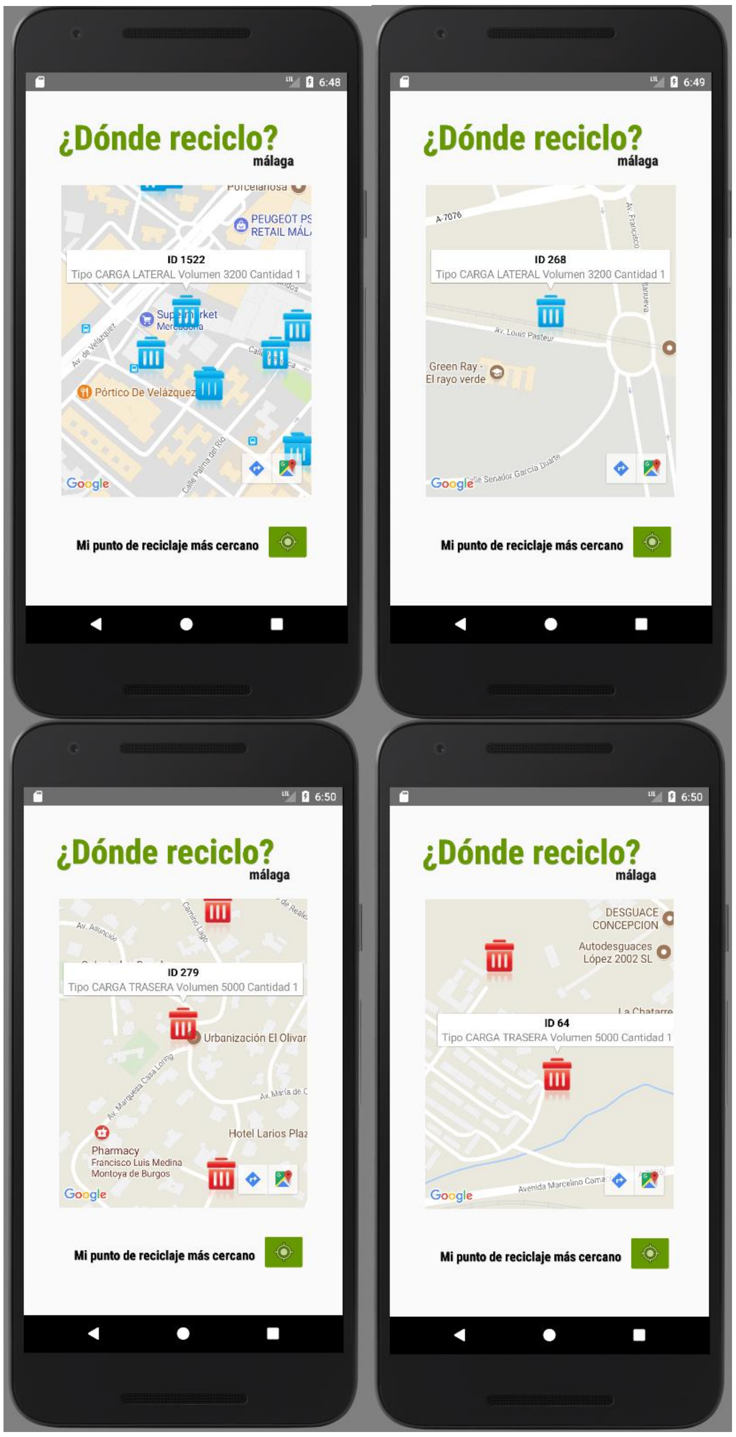

¿Dónde reciclo?
málaga
ID 1912
Tipo CARGA LATERAL Volumen 3200 Cantidad 1
Mi punto de reciclaje más cercano
¿Dónde reciclo?
málaga
ID 1170
Tipo CARGA LATERAL Volumen 3200 Cantidad 1
Mi punto de reciclaje más cercano
¿Dónde reciclo?
málaga
ID 482
Tipo SOTERRADO LATERAL Volumen 3200 Canti...
Mi punto de reciclaje más cercano
¿Dónde reciclo?
málaga
ID 1033
Tipo CAMPANA Volumen 3000 Cantidad 1
Mi punto de reciclaje más cercano

Figure 4: Map Activity for urban solid (grey) and plastic waste (yellow).



ID 1522
Tipo CARGA LATERAL Volumen 3200 Cantidad 1

ID 268
Tipo CARGA LATERAL Volumen 3200 Cantidad 1

ID 279
Tipo CARGA TRASERA Volumen 5000 Cantidad 1

ID 64
Tipo CARGA TRASERA Volumen 5000 Cantidad 1

Figure 5: Map Activity for urban paper (blue) and industrial waste (red).

3. Conclusions

The developed Android application serves as an example of open data app, carrying out basic functions of data visualization and geolocation by using environmental data from the Malaga City Hall open data repository to promote recycling among citizens.

Moreover, the basic scheme of its design and implementation serves a basis to develop other Android applications with different targets which also use datasets in CSV format for their visualizations in Google Maps and related functionalities.

The source code of the java classes of the application can be found in Github repository [5], as well as the external libraries used for its development.

4. References

[1] The Open Definition, http://opendefinition.org/

[2] Portal Datos Abiertos – Ayuntamiento de Málaga, http://datosabiertos.malaga.eu/

[3] Caplan, Robyn, Tim Davies, Asiya Wadud, Stefaan Verhulst, Jose M. Alonso, and Hania Farhan, *"Towards Common Methods for Assessing Open Data: Workshop Report & Draft Framework"*. New York: The Web Foundation and The Gov Lab, 2014.

[4] Yves Grohens, S.Kishor Kumar, Abderrahim Boudenne, Yang Weimin, *"Recycling and Reuse of Materials and Their Products"*. Apple Academic Press, 2013.

[5] Github repository, https://github.com/aracelimanzano

```
package es.uma.aracelimanzano.dondereciclomlg;

import android.net.Uri;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.content.Intent;
import android.os.AsyncTask;
import android.text.style.BulletSpan;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.ImageButton;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.maps.model.LatLng;
import com.opencsv.CSVReader;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.Serializable;
import java.lang.reflect.Array;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.ArrayList;
import java.util.Calendar;

public class MainActivity extends AppCompatActivity {

    private String solidsURL =
"http://datosabiertos.malaga.eu/recursos/ambiente/limasa/cntrsu-
4326.csv";
    private String plasticsURL =
"http://datosabiertos.malaga.eu/recursos/ambiente/limasa/cntenvases-
4326.csv";
    private String paperURL =
"http://datosabiertos.malaga.eu/recursos/ambiente/limasa/cntpapelCarton-
4326.csv";
    private String industrialWasteURL =
"http://datosabiertos.malaga.eu/recursos/ambiente/limasa/cntindustria-
4326.csv";
    private String date;

    public static ArrayList<RecPoint> recyclePointsSolids = new
ArrayList<RecPoint>();
    public static ArrayList<RecPoint> recyclePointsPlastics = new
ArrayList<RecPoint>();
    public static ArrayList<RecPoint> recyclePointsPaper = new
ArrayList<RecPoint>();
    public static ArrayList<RecPoint> recyclePointsIndustrialWaste = new
ArrayList<RecPoint>();

    TextView update;
    ImageButton buttonPlastics;
    ImageButton buttonSolids;
    ImageButton buttonPaper;
    ImageButton buttonIndustrialWaste;

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

        update = (TextView) findViewById(R.id.textViewUpdate);
        buttonPlastics = (ImageButton) findViewById(R.id.buttonPlastics);
        buttonSolids = (ImageButton) findViewById(R.id.buttonSolids);
        buttonPaper = (ImageButton) findViewById(R.id.buttonPaper);
        buttonIndustrialWaste = (ImageButton)
findViewById(R.id.buttonIndustrialWaste);

    }

    public void onClick (View v) throws InterruptedException {

        buttonSolids.setEnabled(false);
        buttonPlastics.setEnabled(false);
        buttonPaper.setEnabled(false);

        switch(v.getId()){

            case R.id.buttonSolids:
                new DownloadCSVTask().execute(solidsURL,"s");
                Thread.sleep(3000);
                if (recyclePointsSolids.isEmpty()){
                    Toast.makeText(getApplicationContext(),
                            "Error de conexión",
Toast.LENGTH_LONG).show();
                }
                else {
                    Intent solidsMap = new Intent(MainActivity.this,
SolidsMapActivity.class);
                    startActivity(solidsMap);}
                break;

            case R.id.buttonPlastics:
                new DownloadCSVTask().execute(plasticsURL,"pl");
                Thread.sleep(3000);
                if (recyclePointsPlastics.isEmpty()){
                    Toast.makeText(getApplicationContext(),
                            "Error de conexión",
Toast.LENGTH_LONG).show();
                }
                else {
                    Intent plasticsMap = new Intent(MainActivity.this,
PlasticsMapActivity.class);
                    startActivity(plasticsMap);}
                break;

            case R.id.buttonPaper:
                new DownloadCSVTask().execute(paperURL,"p");
                Thread.sleep(3000);
                if (recyclePointsPaper.isEmpty()){
                    Toast.makeText(getApplicationContext(),
                            "Error de conexión",
Toast.LENGTH_LONG).show();
                }
                else {
                    Intent paperMap = new Intent(MainActivity.this,
PaperMapActivity.class);
```

```
                    startActivity(paperMap);
                }
                break;

            case R.id.buttonIndustrialWaste:
                new DownloadCSVTask().execute(industrialWasteURL,"ind");
                Thread.sleep(3000);
                if (recyclePointsIndustrialWaste.isEmpty()){
                    Toast.makeText(getApplicationContext(),
                            "", Toast.LENGTH_LONG).show();
                }
                else {
                    Intent industrialWasteMap = new
Intent(MainActivity.this, IndustrialWasteMapActivity.class);
                    startActivity(industrialWasteMap);
                }
                break;

            case R.id.imageWebLimasa:
                startActivity(new Intent(Intent.ACTION_VIEW,
Uri.parse("http://www.limasa3.es/")));
                break;
        }
    }


    private class DownloadCSVTask extends AsyncTask<String,Void,Long>{

        @Override
        protected Long doInBackground(String... strings){

            long res = 0;

            try{
                URL stockURL = new URL(strings[0]);
                BufferedReader in = new BufferedReader(new
InputStreamReader(stockURL.openStream()));
                CSVReader reader = new CSVReader(in);

                String[] nextLine;
                boolean header = true;
                RecPoint recyclePoint;
                String POINT;
                String geos [];
                int id = 0;

                while((nextLine = reader.readNext()) != null){

                    if (header){
                        header = false;
                    }

                    else{
                        recyclePoint = new
RecPoint(id,nextLine[2],Integer.parseInt(nextLine[3]),Integer.parseInt(ne
xtLine[4]));

                        POINT = nextLine[7];
                        POINT=POINT.replace('(',' ');
                        POINT=POINT.replace(')',' ');
                        geos = POINT.split(" ");

recyclePoint.setLatLng(Double.parseDouble(geos[3]),
Double.parseDouble(geos[2]));

                        recyclePoint.setAvailableUpdate(nextLine[5]);

                        if (strings[1].equals("s")) {
                            recyclePointsSolids.add(recyclePoint);
                        }
                        else if (strings[1].equals("pl")){
                            recyclePointsPlastics.add(recyclePoint);
                        }
                        else if (strings[1].equals("p")){
                            recyclePointsPaper.add(recyclePoint);
                        }
                        else if (strings[1].equals("ind")){

recyclePointsIndustrialWaste.add(recyclePoint);
                        }
                        else{
                            res = -1;
                            throw new RecPointException();
                        }

                        id++;
                    }
                }

                res = 1;
            }

            catch (MalformedURLException mue){
                res = -1;
                throw new RecPointException();
            }

            catch (IOException ioe){
                res = -1;
                throw new RecPointException();
            }

            return res;
        }

        @Override
        protected void onPostExecute(Long res){

            buttonSolids.setEnabled(true);
            buttonPlastics.setEnabled(true);
            buttonPaper.setEnabled(true);

            if (res==1){

                Calendar rightNow = Calendar.getInstance();
                date = rightNow.getTime().toString();
                update.setText("Última actualización: "+ date);
            }

            else {
                update.setText("Error de conexión");
            }
        }
    }

}
```

```java
package es.uma.aracelimanzano.dondereciclomlg;

import android.support.v4.app.ActivityCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.Manifest;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.pm.PackageManager;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.BaseAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.MapFragment;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.model.BitmapDescriptor;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import java.util.ArrayList;

public class IndustrialWasteMapActivity extends AppCompatActivity
implements OnMapReadyCallback {

    public static final int REQUEST_LOCATION_PER = 1;
    public static final String PREF_NAME = "GPS";
    public static final String LAT_NAME = "Latitude";
    public static final String LNG_NAME = "Longitude";

    GoogleMap industrialWasteMap = null;
    ArrayList<RecPoint> recyclePoints;
    LocationListener locationListener;
    LocationManager locationManager;
    Location currentLocation;

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

        SharedPreferences settings = getSharedPreferences(PREF_NAME, 0);
        String latString = settings.getString(LAT_NAME,"");
        String lngString = settings.getString(LNG_NAME,"");

        currentLocation = new Location("Default");
        currentLocation.setLatitude(36.71853911463124);
        currentLocation.setLongitude(-4.496980905532837);
        locationManager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);
        locationListener = new LocationListenerRecPoint();

        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION)
                != PackageManager.PERMISSION_GRANTED) {

            ActivityCompat.requestPermissions(this,
                    new
String[]{Manifest.permission.ACCESS_COARSE_LOCATION},
                    REQUEST_LOCATION_PER);
        } else {

            Location locationNetwork =

locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
            if (locationNetwork != null)
                currentLocation = locationNetwork;
        }

        recyclePoints = MainActivity.recyclePointsIndustrialWaste;

        MapFragment paperMap = (MapFragment)
getFragmentManager().findFragmentById(R.id.industrialWasteMap);
        paperMap.getMapAsync(this);
    }

    @Override
    public void onMapReady(GoogleMap googleMap) {

        industrialWasteMap = googleMap;

        for (RecPoint rp : recyclePoints) {

            LatLng rpLatLng = new LatLng(rp.getLatitude(),
rp.getLongitude());
            BitmapDescriptor icon =
BitmapDescriptorFactory.fromResource(R.drawable.red_bin);

            industrialWasteMap.addMarker(new MarkerOptions()
                    .title("ID " + rp.getIDString())
                    .snippet("Tipo " + rp.getType() + " Volumen " +
rp.getVolume() + " Cantidad " + rp.getQuantity())
                    .position(rpLatLng)
                    .icon(icon));
        }

        int zoom = 17;
        moveCamera(recyclePoints.get(0).getLatitude(),
recyclePoints.get(0).getLongitude(), zoom);

    }


    private void moveCamera(double lat, double lng, int zoom) {

        LatLng point = new LatLng(lat, lng);
```

```
industrialWasteMap.moveCamera(CameraUpdateFactory.newLatLngZoom(point,
zoom));
    }

    private Location convert(RecPoint rp) {

        Location rpLocation = new Location(rp.getIDString());

        rpLocation.setLongitude(rp.getLongitude());
        rpLocation.setLatitude(rp.getLatitude());

        return rpLocation;
    }

    public void onClick(View v) {

        if (v.getId() == R.id.buttonNearest) {


            float distance = Float.MAX_VALUE, aux;
            int c = 0, selected = -1;
            RecPoint selectedRecyclePoint;
            Location rpLocation;

            for (RecPoint rp : recyclePoints) {

                rpLocation = convert(rp);
                aux = currentLocation.distanceTo(rpLocation);

                if (aux < distance) {
                    distance = aux;
                    selected = c;
                }

                c++;
            }

            RecPoint select = recyclePoints.get(selected);

            if (industrialWasteMap != null) {
                int zoom = 17;
                moveCamera(select.getLatitude(), select.getLongitude(),
zoom);
            }
        }
    }

    protected void onStop(){
        super.onStop();
        SharedPreferences settings = getSharedPreferences(PREF_NAME,0);
        SharedPreferences.Editor editor = settings.edit();

        for (RecPoint rp: recyclePoints) {
            editor.putString("",rp.getIDString());
        }


editor.putString(LAT_NAME,Double.toString(currentLocation.getLatitude()))
;
```

```
editor.putString(LNG_NAME,Double.toString(currentLocation.getLongitude())
);

        editor.commit();
    }

    private class LocationListenerRecPoint implements LocationListener {

        @Override
        public void onLocationChanged(Location location) {

            currentLocation = location;

            Toast.makeText(getApplicationContext(),
                    "Location has changed: " + location.getLatitude() + "
" + location.getLongitude(),
                    Toast.LENGTH_SHORT).show();

            SharedPreferences settings =
getSharedPreferences(PREF_NAME,0);
            SharedPreferences.Editor editor = settings.edit();


editor.putString(LAT_NAME,Double.toString(currentLocation.getLatitude()))
;

editor.putString(LNG_NAME,Double.toString(currentLocation.getLongitude())
);
        }

        @Override
        public void onStatusChanged(String s, int i, Bundle bundle) {}

        @Override
        public void onProviderEnabled(String s) {}

        @Override
        public void onProviderDisabled(String s) {}
    }

    public void onRequestPermissionsResult(int requestCode, String
permisssions[], int[] grantResults) {

        switch (requestCode) {
            case REQUEST_LOCATION_PER:

                if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                    Toast.makeText(getApplicationContext(),
                            "Permisos
Concedidos",Toast.LENGTH_SHORT).show();

                    long minTime      = 5000;  //5 sg
                    float minDistance = 1000;  //1 km

                    if (ActivityCompat.checkSelfPermission(this,
                            Manifest.permission.ACCESS_COARSE_LOCATION)
!= PackageManager.PERMISSION_GRANTED) {
```

```
                        ActivityCompat.requestPermissions(this,
                                new
String[]{Manifest.permission.ACCESS_COARSE_LOCATION},
                                REQUEST_LOCATION_PER);

                        Toast.makeText(getApplicationContext(),
                                "Error", Toast.LENGTH_SHORT).show();
                    }

                    else {

                        ActivityCompat.requestPermissions(this,
                                new
String[]{Manifest.permission.ACCESS_COARSE_LOCATION},
                                REQUEST_LOCATION_PER);

                        Location locationNetwork =

locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
                        if (locationNetwork!=null)
                            currentLocation = locationNetwork;

                        locationManager.requestLocationUpdates(
                                LocationManager.NETWORK_PROVIDER,
                                minTime,
                                minDistance,
                                locationListener);
                    }
                }

                break;
        }
    }
}
```

```
package es.uma.aracelimanzano.dondereciclomlg;

import android.support.v4.app.ActivityCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.Manifest;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.pm.PackageManager;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.BaseAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.MapFragment;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.model.BitmapDescriptor;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import java.util.ArrayList;

public class PaperMapActivity extends AppCompatActivity implements
OnMapReadyCallback {

    public static final int REQUEST_LOCATION_PER = 1;
    public static final String PREF_NAME = "GPS";
    public static final String LAT_NAME = "Latitude";
    public static final String LNG_NAME = "Longitude";

    GoogleMap paperMap = null;
    ArrayList<RecPoint> recyclePoints;
    LocationListener locationListener;
    LocationManager locationManager;
    Location currentLocation;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_paper_map);
        SharedPreferences settings = getSharedPreferences(PREF_NAME, 0);
        String latString = settings.getString(LAT_NAME,"");
        String lngString = settings.getString(LNG_NAME,"");

        currentLocation = new Location("Default");
        currentLocation.setLatitude(36.71853911463124);
        currentLocation.setLongitude(-4.496980905532837);
        locationManager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);
        locationListener = new LocationListenerRecPoint();

        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION)
                != PackageManager.PERMISSION_GRANTED) {

            ActivityCompat.requestPermissions(this,
                    new
String[]{Manifest.permission.ACCESS_COARSE_LOCATION},
                    REQUEST_LOCATION_PER);
        } else {

            Location locationNetwork =

locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
            if (locationNetwork != null)
                currentLocation = locationNetwork;
        }

        recyclePoints = MainActivity.recyclePointsPaper;

        MapFragment paperMap = (MapFragment)
getFragmentManager().findFragmentById(R.id.paperMap);
        paperMap.getMapAsync(this);
    }

    @Override
    public void onMapReady(GoogleMap googleMap) {

        paperMap = googleMap;

        for (RecPoint rp : recyclePoints) {

            LatLng rpLatLng = new LatLng(rp.getLatitude(),
rp.getLongitude());
            BitmapDescriptor icon =
BitmapDescriptorFactory.fromResource(R.drawable.blue_bin);
            paperMap.addMarker(new MarkerOptions()
                    .title("ID " + rp.getIDString())
                    .snippet("Tipo " + rp.getType() + " Volumen " +
rp.getVolume() + " Cantidad " + rp.getQuantity())
                    .position(rpLatLng)
                    .icon(icon));
        }

        int zoom = 17;

        Log.d("UNO",recyclePoints.get(0).getType());

        moveCamera(recyclePoints.get(0).getLatitude(),
recyclePoints.get(0).getLongitude(), zoom);

    }


    private void moveCamera(double lat, double lng, int zoom) {

        LatLng point = new LatLng(lat, lng);
        paperMap.moveCamera(CameraUpdateFactory.newLatLngZoom(point,
zoom));
    }

    private Location convert(RecPoint rp) {

        Location rpLocation = new Location(rp.getIDString());

        rpLocation.setLongitude(rp.getLongitude());
        rpLocation.setLatitude(rp.getLatitude());

        return rpLocation;
    }

    public void onClick(View v) {

        if (v.getId() == R.id.buttonNearest) {


            float distance = Float.MAX_VALUE, aux;
            int c = 0, selected = -1;
            RecPoint selectedRecyclePoint;
            Location rpLocation;

            for (RecPoint rp : recyclePoints) {

                rpLocation = convert(rp);
                aux = currentLocation.distanceTo(rpLocation);

                if (aux < distance) {
                    distance = aux;
                    selected = c;
                }

                c++;
            }

            RecPoint select = recyclePoints.get(selected);

            if (paperMap != null) {
                int zoom = 17;
                moveCamera(select.getLatitude(), select.getLongitude(),
zoom);
            }
        }
    }

    protected void onStop(){
        super.onStop();
        SharedPreferences settings = getSharedPreferences(PREF_NAME,0);
        SharedPreferences.Editor editor = settings.edit();

        for (RecPoint rp: recyclePoints) {
            editor.putString("",rp.getIDString());
        }


editor.putString(LAT_NAME,Double.toString(currentLocation.getLatitude()))
;
```

```
editor.putString(LNG_NAME,Double.toString(currentLocation.getLongitude())
);

        editor.commit();
    }

    private class LocationListenerRecPoint implements LocationListener {

        @Override
        public void onLocationChanged(Location location) {

            currentLocation = location;

            Toast.makeText(getApplicationContext(),
                    "Location has changed: " + location.getLatitude() + "
" + location.getLongitude(),
                    Toast.LENGTH_SHORT).show();

            SharedPreferences settings =
getSharedPreferences(PREF_NAME,0);
            SharedPreferences.Editor editor = settings.edit();


editor.putString(LAT_NAME,Double.toString(currentLocation.getLatitude()))
;

editor.putString(LNG_NAME,Double.toString(currentLocation.getLongitude())
);
        }

        @Override
        public void onStatusChanged(String s, int i, Bundle bundle) {}

        @Override
        public void onProviderEnabled(String s) {}

        @Override
        public void onProviderDisabled(String s) {}
    }

    public void onRequestPermissionsResult(int requestCode, String
permisssions[], int[] grantResults) {

        switch (requestCode) {
            case REQUEST_LOCATION_PER:

                if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                    Toast.makeText(getApplicationContext(),
                            "Permisos
Concedidos",Toast.LENGTH_SHORT).show();

                    long minTime      = 5000;  //5 sg
                    float minDistance = 1000;  //1 km

                    if (ActivityCompat.checkSelfPermission(this,
                            Manifest.permission.ACCESS_COARSE_LOCATION)
!= PackageManager.PERMISSION_GRANTED) {
                        Toast.makeText(getApplicationContext(),
                                "Error", Toast.LENGTH_SHORT).show();

                        ActivityCompat.requestPermissions(this,
                                new
String[]{Manifest.permission.ACCESS_COARSE_LOCATION},
                                REQUEST_LOCATION_PER);
                    }

                    else {

                        Location locationNetwork =

locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
                        if (locationNetwork!=null)
                            currentLocation = locationNetwork;


                        locationManager.requestLocationUpdates(
                                LocationManager.NETWORK_PROVIDER,
                                minTime,
                                minDistance,
                                locationListener);
                    }
                }

                break;
        }

    }

}
```

```java
package es.uma.aracelimanzano.dondereciclomlg;

import android.support.v4.app.ActivityCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.Manifest;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.pm.PackageManager;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.BaseAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.MapFragment;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.model.BitmapDescriptor;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import java.util.ArrayList;

public class PlasticsMapActivity extends AppCompatActivity implements
OnMapReadyCallback {

    public static final int REQUEST_LOCATION_PER = 1;
    public static final String PREF_NAME = "GPS";
    public static final String LAT_NAME = "Latitude";
    public static final String LNG_NAME = "Longitude";

    GoogleMap plasticsMap = null;
    ArrayList<RecPoint> recyclePoints;
    LocationListener locationListener;
    LocationManager locationManager;
    Location currentLocation;

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

        SharedPreferences settings = getSharedPreferences(PREF_NAME, 0);
        String latString = settings.getString(LAT_NAME,"");
        String lngString = settings.getString(LNG_NAME,"");

        currentLocation = new Location("Default");
        currentLocation.setLatitude(36.71853911463124);
        currentLocation.setLongitude(-4.496980905532837);
        locationManager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);
        locationListener = new LocationListenerRecPoint();

        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION)
                != PackageManager.PERMISSION_GRANTED) {

            ActivityCompat.requestPermissions(this,
                    new
String[]{Manifest.permission.ACCESS_COARSE_LOCATION},
                    REQUEST_LOCATION_PER);
        } else {

            Location locationNetwork =

locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
            if (locationNetwork != null)
                currentLocation = locationNetwork;
        }

        recyclePoints = MainActivity.recyclePointsPlastics;

        MapFragment paperMap = (MapFragment)
getFragmentManager().findFragmentById(R.id.plasticsMap);
        paperMap.getMapAsync(this);
    }

    @Override
    public void onMapReady(GoogleMap googleMap) {

        plasticsMap = googleMap;

        for (RecPoint rp : recyclePoints) {

            LatLng rpLatLng = new LatLng(rp.getLatitude(),
rp.getLongitude());
            BitmapDescriptor icon =
BitmapDescriptorFactory.fromResource(R.drawable.yellow_bin);
            plasticsMap.addMarker(new MarkerOptions()
                    .title("ID " + rp.getIDString())
                    .snippet("Tipo " + rp.getType() + " Volumen " +
rp.getVolume() + " Cantidad " + rp.getQuantity())
                    .position(rpLatLng)
                    .icon(icon));
        }

        int zoom = 17;
        moveCamera(recyclePoints.get(0).getLatitude(),
recyclePoints.get(0).getLongitude(), zoom);

    }


    private void moveCamera(double lat, double lng, int zoom) {

        LatLng point = new LatLng(lat, lng);
        plasticsMap.moveCamera(CameraUpdateFactory.newLatLngZoom(point,
zoom));
    }
    private Location convert(RecPoint rp) {

        Location rpLocation = new Location(rp.getIDString());

        rpLocation.setLongitude(rp.getLongitude());
        rpLocation.setLatitude(rp.getLatitude());

        return rpLocation;
    }

    public void onClick(View v) {

        if (v.getId() == R.id.buttonNearest) {


            float distance = Float.MAX_VALUE, aux;
            int c = 0, selected = -1;
            RecPoint selectedRecyclePoint;
            Location rpLocation;

            for (RecPoint rp : recyclePoints) {

                rpLocation = convert(rp);
                aux = currentLocation.distanceTo(rpLocation);

                if (aux < distance) {
                    distance = aux;
                    selected = c;
                }

                c++;
            }

            RecPoint select = recyclePoints.get(selected);

            if (plasticsMap != null) {
                int zoom = 17;
                moveCamera(select.getLatitude(), select.getLongitude(),
zoom);
            }
        }
    }

    protected void onStop(){
        super.onStop();
        SharedPreferences settings = getSharedPreferences(PREF_NAME,0);
        SharedPreferences.Editor editor = settings.edit();

        for (RecPoint rp: recyclePoints) {
            editor.putString("",rp.getIDString());
        }


editor.putString(LAT_NAME,Double.toString(currentLocation.getLatitude()))
;

editor.putString(LNG_NAME,Double.toString(currentLocation.getLongitude())
);
```

```
        editor.commit();
    }

    private class LocationListenerRecPoint implements LocationListener {

        @Override
        public void onLocationChanged(Location location) {

            currentLocation = location;

            Toast.makeText(getApplicationContext(),
                    "Location has changed: " + location.getLatitude() + "
" + location.getLongitude(),
                    Toast.LENGTH_SHORT).show();

            SharedPreferences settings =
getSharedPreferences(PREF_NAME,0);
            SharedPreferences.Editor editor = settings.edit();


editor.putString(LAT_NAME,Double.toString(currentLocation.getLatitude()))
;

editor.putString(LNG_NAME,Double.toString(currentLocation.getLongitude())
);
        }

        @Override
        public void onStatusChanged(String s, int i, Bundle bundle) {}

        @Override
        public void onProviderEnabled(String s) {}

        @Override
        public void onProviderDisabled(String s) {}
    }

    public void onRequestPermissionsResult(int requestCode, String
permisssions[], int[] grantResults) {

        switch (requestCode) {
            case REQUEST_LOCATION_PER:

                if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                    Toast.makeText(getApplicationContext(),
                            "Permisos
Concedidos",Toast.LENGTH_SHORT).show();

                    long minTime      = 5000;  //5 sg
                    float minDistance = 1000;  //1 km

                    if (ActivityCompat.checkSelfPermission(this,
                            Manifest.permission.ACCESS_COARSE_LOCATION)
!= PackageManager.PERMISSION_GRANTED) {

                        ActivityCompat.requestPermissions(this,
                                new
String[]{Manifest.permission.ACCESS_COARSE_LOCATION},
                                REQUEST_LOCATION_PER);
                        Toast.makeText(getApplicationContext(),
                                "Error", Toast.LENGTH_SHORT).show();
                    }

                    else {

                        ActivityCompat.requestPermissions(this,
                                new
String[]{Manifest.permission.ACCESS_COARSE_LOCATION},
                                REQUEST_LOCATION_PER);

                        Location locationNetwork =

locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
                        if (locationNetwork!=null)
                            currentLocation = locationNetwork;

                        locationManager.requestLocationUpdates(
                                LocationManager.NETWORK_PROVIDER,
                                minTime,
                                minDistance,
                                locationListener);
                    }
                }

                break;
        }
    }
}
```

```
package es.uma.aracelimanzano.dondereciclomlg;

import android.support.v4.app.ActivityCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.Manifest;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.pm.PackageManager;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.BaseAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.common.data.DataHolder;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.MapFragment;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.model.BitmapDescriptor;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import com.opencsv.CSVReader;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.ArrayList;

public class SolidsMapActivity extends AppCompatActivity implements
OnMapReadyCallback {

    public static final int REQUEST_LOCATION_PER = 1;
    public static final String PREF_NAME = "GPS";
    public static final String LAT_NAME = "Latitude";
    public static final String LNG_NAME = "Longitude";

    GoogleMap solidsMap = null;
    ArrayList<RecPoint> recyclePoints;
    LocationListener locationListener;
    LocationManager locationManager;
    Location currentLocation;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_solids_map);
        SharedPreferences settings = getSharedPreferences(PREF_NAME, 0);
        String latString = settings.getString(LAT_NAME,"");
        String lngString = settings.getString(LNG_NAME,"");

        currentLocation = new Location("Default");
        currentLocation.setLatitude(36.71853911463124);
        currentLocation.setLongitude(-4.496980905532837);

        locationManager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);
        locationListener = new LocationListenerRecPoint();

        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION)
                != PackageManager.PERMISSION_GRANTED) {

            ActivityCompat.requestPermissions(this,
                    new
String[]{Manifest.permission.ACCESS_COARSE_LOCATION},
REQUEST_LOCATION_PER);
        } else {

            Location locationNetwork =

locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
            if (locationNetwork != null)
                currentLocation = locationNetwork;
        }

        recyclePoints = MainActivity.recyclePointsSolids;

        MapFragment solidsMap = (MapFragment)
getFragmentManager().findFragmentById(R.id.solidsMap);
        solidsMap.getMapAsync(this);
    }

    @Override
    public void onMapReady(GoogleMap googleMap) {

        solidsMap = googleMap;

        for (RecPoint rp : recyclePoints) {

            LatLng rpLatLng = new LatLng(rp.getLatitude(),
rp.getLongitude());
            BitmapDescriptor icon =
BitmapDescriptorFactory.fromResource(R.drawable.grey_bin);
            solidsMap.addMarker(new MarkerOptions()
                    .title("ID " + rp.getIDString())
                    .snippet("Tipo " + rp.getType() + " Volumen " +
rp.getVolume() + " Cantidad " + rp.getQuantity())
                    .position(rpLatLng)
                    .icon(icon));
        }

        int zoom = 17;

        Log.d("UNO",recyclePoints.get(0).getType());
        moveCamera(recyclePoints.get(0).getLatitude(),
recyclePoints.get(0).getLongitude(), zoom);

    }


    private void moveCamera(double lat, double lng, int zoom) {

        LatLng point = new LatLng(lat, lng);
        solidsMap.moveCamera(CameraUpdateFactory.newLatLngZoom(point,
zoom));
    }

    private Location convert(RecPoint rp) {

        Location rpLocation = new Location(rp.getIDString());

        rpLocation.setLongitude(rp.getLongitude());
        rpLocation.setLatitude(rp.getLatitude());

        return rpLocation;
    }

    public void onClick(View v) {

        if (v.getId() == R.id.buttonNearest) {


            float distance = Float.MAX_VALUE, aux;
            int c = 0, selected = -1;
            RecPoint selectedRecyclePoint;
            Location rpLocation;

            for (RecPoint rp : recyclePoints) {

                rpLocation = convert(rp);
                aux = currentLocation.distanceTo(rpLocation);

                if (aux < distance) {
                    distance = aux;
                    selected = c;
                }

                c++;
            }

            RecPoint select = recyclePoints.get(selected);

            if (solidsMap != null) {
                int zoom = 17;
                moveCamera(select.getLatitude(), select.getLongitude(),
zoom);
            }
        }
    }

    protected void onStop(){
        super.onStop();
        SharedPreferences settings = getSharedPreferences(PREF_NAME,0);
        SharedPreferences.Editor editor = settings.edit();
        for (RecPoint rp: recyclePoints) {
            editor.putString("",rp.getIDString());
        }


editor.putString(LAT_NAME,Double.toString(currentLocation.getLatitude()))
;

editor.putString(LNG_NAME,Double.toString(currentLocation.getLongitude())
);

        editor.commit();
    }

    private class LocationListenerRecPoint implements LocationListener {

        @Override
        public void onLocationChanged(Location location) {

            currentLocation = location;

            Toast.makeText(getApplicationContext(),
                    "Location has changed: " + location.getLatitude() + "
" + location.getLongitude(),
                    Toast.LENGTH_SHORT).show();

            SharedPreferences settings =
getSharedPreferences(PREF_NAME,0);
            SharedPreferences.Editor editor = settings.edit();


editor.putString(LAT_NAME,Double.toString(currentLocation.getLatitude()))
;

editor.putString(LNG_NAME,Double.toString(currentLocation.getLongitude())
);
        }

        @Override
        public void onStatusChanged(String s, int i, Bundle bundle) {}

        @Override
        public void onProviderEnabled(String s) {}

        @Override
        public void onProviderDisabled(String s) {}
    }

    public void onRequestPermissionsResult(int requestCode, String
permisssions[], int[] grantResults) {

        switch (requestCode) {
            case REQUEST_LOCATION_PER:

                if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                    Toast.makeText(getApplicationContext(),
                            "Permisos
Concedidos",Toast.LENGTH_SHORT).show();
```

```
                    long minTime      = 5000;  //5 sg
                    float minDistance = 1000;  //1 km

                    if (ActivityCompat.checkSelfPermission(this,
                            Manifest.permission.ACCESS_COARSE_LOCATION)
!= PackageManager.PERMISSION_GRANTED) {
                        Toast.makeText(getApplicationContext(),
                                "Error", Toast.LENGTH_SHORT).show();

                        ActivityCompat.requestPermissions(this,
                                new
String[]{Manifest.permission.ACCESS_COARSE_LOCATION},
                                REQUEST_LOCATION_PER);
                    }

                    else {

                        Location locationNetwork =

locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
                        if (locationNetwork!=null)
                            currentLocation = locationNetwork;


                        locationManager.requestLocationUpdates(
                                LocationManager.NETWORK_PROVIDER,
                                minTime,
                                minDistance,
                                locationListener);
                    }
                }

                break;
        }

    }
}
```

```java
package es.uma.aracelimanzano.dondereciclomlg;

/**
 * Created by aracelimanzano on 8/11/17.
 */

public class RecPoint {

    private String _type = "Non specified type";
    private int _fid = -1;
    private int _v = -1;
    private int _q = -1;
    private double _latitude = 0.0;
    private double _longitude = 0.0;
    private String _avUpdate = "Non available update";

    public RecPoint(int fid, String type, int v, int q){
        _fid = fid;
        _type = type;
        _v = v;
        _q = q;
    }

    public void setLatLng (double latitude, double longitude){
        _latitude = latitude;
        _longitude = longitude;
    }

    public void setAvailableUpdate (String avUpdate){
        _avUpdate = avUpdate;
    }

    public String getIDString() { return ""+_fid; }

    public String getType(){
        return _type;
    }

    public int getVolume(){
        return _v;
    }

    public int getQuantity(){
        return _q;
    }

    public Double getLatitude(){
        return _latitude;
    }

    public Double getLongitude(){
        return _longitude;
    }

    public String getAvailableUpdate(){
        return _avUpdate;
    }


}
```

```
package es.uma.aracelimanzano.dondereciclomlg;

/**
 * Created by aracelimanzano on 8/11/17.
 */

public class RecPointException extends RuntimeException {

    public RecPointException(){}
    public RecPointException(String msg){ super(msg); }

}
```