

---

# Mobile Application Development

## Google Maps Android API v2

Waterford Institute of Technology

November 5, 2014

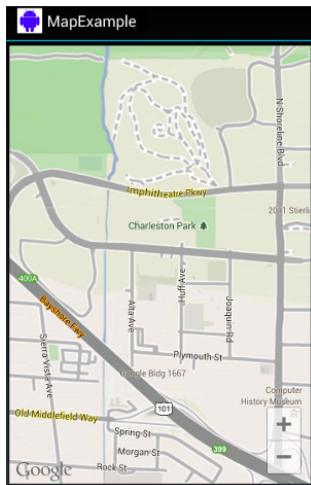
John Fitzgerald

# Google Maps Android API v2

## API features

### Google Maps Android API

- Embed & display map
- Access Google Map servers
- Download map data
- Add markers, polygons, overlays
- Change zoom level
- Determine geolocation
- Select map type (normal, hybrid ...)

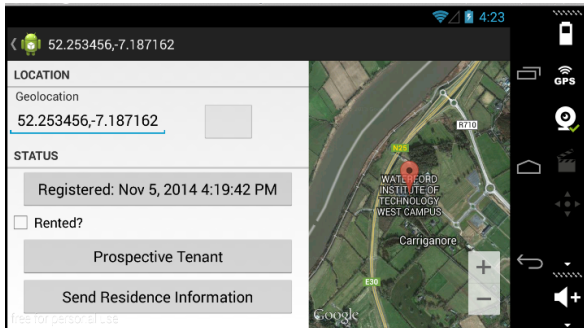


# Google Maps Android API v2

## Preparatory work

### Google Play Services

- Install via Android SDK Manager
- Import to MyRent workspace
- Reference in manifest file
- Reference in MyRent properties



# Google Maps Android API v2

## Maps API key

### **Generating & using key**

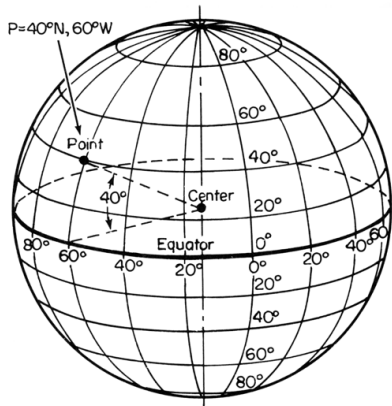
- Each app requires key
- Key obtainable at Google API console
- Must be registered user
- Generate key
- Add key to manifest
- Switch on API v2 in console

# Google Maps Android API v2

## Geolocation

### LatLng stores map data

- Units are degrees
- Ranges:
  - Longitude: 0 to +/- 180
  - Latitude: 0 to +/- 90
- One degree (metres)
  - latitude approx 111
  - longitude same at equator
  - zero at poles
- Accuracy decimal places:
  - Four: 11 m (Garmin 15 m)
  - Six: 11 cm
  - Seven: 11 mm



# Google Maps Android API v2

## Helpers

### Data input and manipulation

- Geolocation data input as String
- Necessary convert to & from Android **LatLng**

```
public static LatLng latLng(Context context, String geolocation)
{
    String[] g = geolocation.split(",");
    return new LatLng(Double.parseDouble(g[0]), Double.parseDouble(g[1]));
}
```

```
public static String latLng(LatLng geo)
{
    return String.format("%.6f", geo.latitude) + ", "
        + String.format("%.6f", % geo.longitude);
}
```

# Google Maps Android API v2

## Modify manifest file

- Add permissions

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<!-- The following two permissions are not required to use
     Google Maps Android API v2, but are recommended. -->
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
```

- Add API key

```
<meta-data
  android:name="com.google.android.maps.v2.API_KEY"
  android:value="TODO: Insert your API key here" />

<meta-data
  android:name="com.google.android.gms.version"
  android:value="@integer/google_play_services_version"/>
```

# Google Maps Android API v2

## ResidenceFragment

### Fields added to ResidenceFragment

```
SupportMapFragment mapFragment;
```

```
GoogleMap gmap;
```

```
Marker marker;
```

```
LatLng markerPosition;
```

```
boolean markerDragged;
```

Wrapper around view of map

Main Google Maps class

Icon placed at geolocation

Contains geolocation

Indicates how marker moved



# Google Maps Android API v2

## ResidenceFragment

### Initialize map fragment in **onActivityCreated**

```
private void initializeMapFragment()
{
    FragmentManager fm = getChildFragmentManager();
    mapFragment = (SupportMapFragment) fm.findFragmentById(R.id.map);
    if (mapFragment == null)
    {
        mapFragment = SupportMapFragment.newInstance();
        fm.beginTransaction().replace(R.id.map, mapFragment).commit();
    }
}
```

# Google Maps Android API v2

## ResidenceFragment

### Implement interfaces

```
public class ResidenceFragment extends SupportMapFragment implements TextWatcher,  
    OnCheckedChangeListener,  
    OnClickListener,  
    DatePickerDialog.OnDateSetListener,  
    GoogleMap.OnMarkerDragListener,  
    GoogleMap.OnCameraChangeListener
```

Listens for marker dragged

Listens for changes to map camera

# Google Maps Android API v2

## ResidenceFragment

### OnMarkerDragListener

- Three methods to implement
- Fully implement only **onMarkerDragEnd**

```
@Override
public void onMarkerDragEnd(Marker arg0)
{
    residence.geolocation = MapHelper.latLng(arg0.getPosition());
    getActivity().setTitle(residence.geolocation);
    gmap.animateCamera(CameraUpdateFactory.newLatLng(arg0.getPosition()));
    markerDragged = true;
}
```

# Google Maps Android API v2

## ResidenceFragment

### OnCameraChangeListener

- Triggered by pan, zoom ...

```
@Override
public void onCameraChange(CameraPosition arg0)
{
    residence.zoom = arg0.zoom;
    markerPosition = MapHelper.latLng(getActivity(), residence.geolocation);
    if (marker != null)
    {
        marker.remove();
        marker = null;
    }
    MarkerOptions o = new MarkerOptions()
        .position(markerPosition)
        .draggable(true).title("Residence")
        .alpha(0.7f)
        .snippet("GPS : " + markerPosition.toString());
    marker = gmap.addMarker(o);
}
```

# Google Maps Android API v2

## ResidenceFragment

### Initialize Map when fragment starts

- Invoke in **onStart**
- Register marker & camera listeners
- Set map type (e.g. Hybrid, Terrain)
- Display map

```
private void initializeMap(LatLng markerPosition)
{
    if (mapFragment != null) {
        gmap = mapFragment.getMap();
        if (gmap != null) {
            gmap.animateCamera(CameraUpdateFactory.newLatLngZoom(markerPosition,
                (float) residence.zoom));
            gmap.setMapType(GoogleMap.MAP_TYPE_HYBRID);
            gmap.setOnMarkerDragListener(this);
            gmap.setOnCameraChangeListener(this);
        }
    }
}
```

# Google Maps Android API v2

## ResidenceFragment

### Update Map

- When geolocation changed manually (**afterTextChanged**)
- Boolean flag used to distinguish manual geolocation changes and those caused by dragging marker.

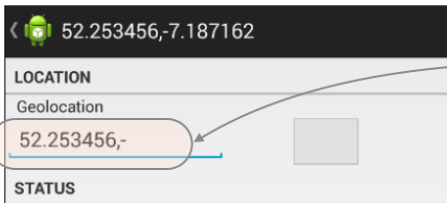
```
private void updateMap(LatLng markerPosition)
{
    if (mapFragment != null)
    {
        gmap = mapFragment.getMap();
        if (gmap != null) {
            gmap.animateCamera(CameraUpdateFactory.newLatLngZoom(markerPosition,
                (float) residence.zoom));
        }
    }
}
```

# Google Maps Android API v2

## ResidenceFragment

### Geolocation data input

- Validation introduced in MapHelper.latlng
- Disallows invalid geolocation
- try-catch block used



Example invalid data input