ΤΕΙ Αθήνας – Σ.Τ.ΕΦ – Τμ. Μηχανικών Πληροφορικής ΤΕ
M.sc στις Τεχνολογίες υπολογισμού & δικτύων - Ιούνιος 2017

Δίκτυα Κινητών Επικοινωνιών και Εφαρμογές

Ανάπτυξη Android εφαρμογής
η οποία αναδεικνύει τις τεχνολογίες γύρω από τα δίκτυα κινητών επικοινωνιών.

Παναγιώτης Μαυροδάκος (16010) & Μιχαήλ Γαλλιάκης (16003)

Διδάσκων καθηγητής:
Αντώνης Μπόγρης

Στόχος εργασίας

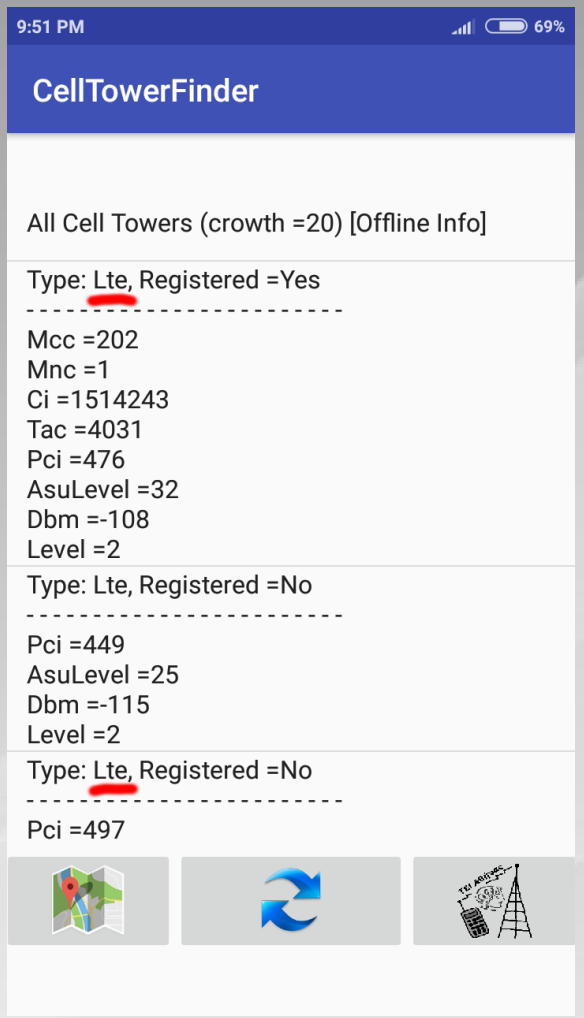
- Εύρεση συνδεδεμένου σταθμού βάσης
- Αποθήκευση σταθμού σε DB
- Εμφάνιση cell tower στον χάρτη
- Χαρτογράφηση

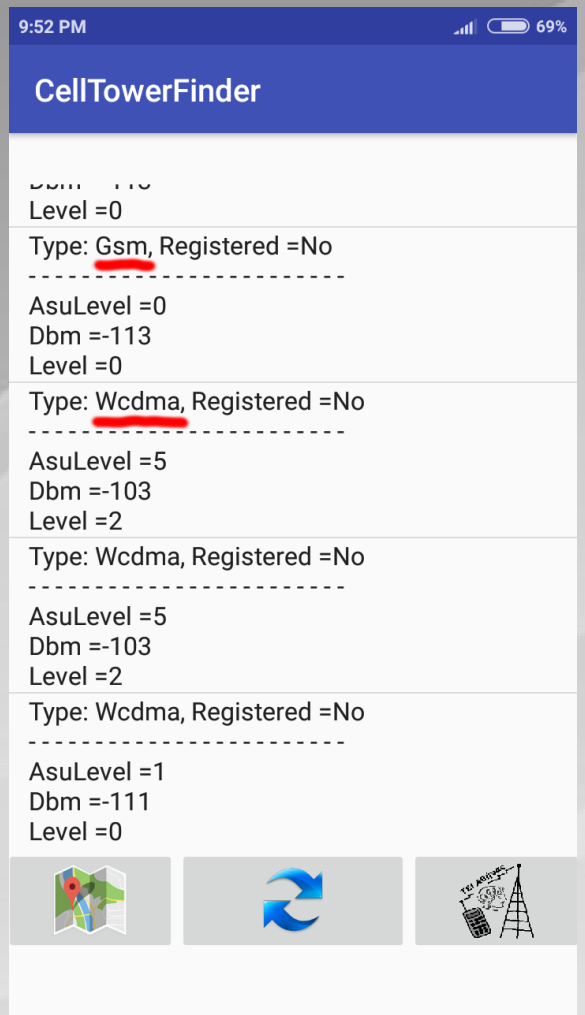
Βασικές ενότητες

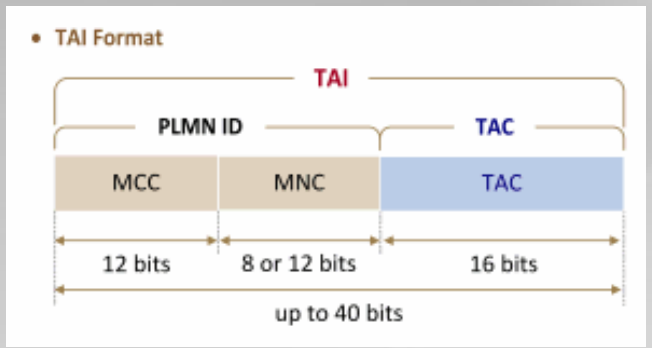
- Παρουσίαση εφαρμογής
- Τεχνικά χαρακτηριστικά
- Troubleshooting
- Μελλοντικές επεκτάσεις
- Συμπεράσματα

Παρουσίαση εφαρμογής

Παρουσίαση εφαρμογής



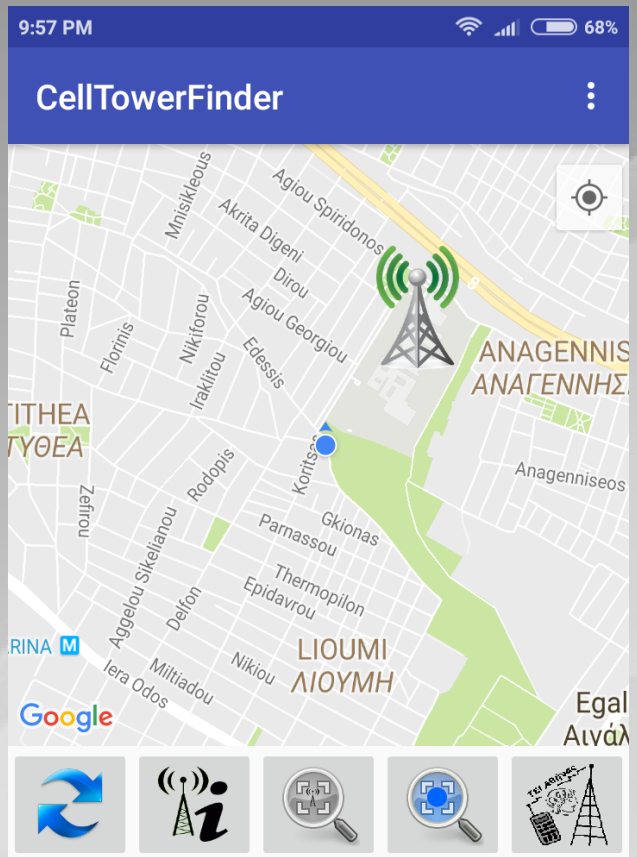


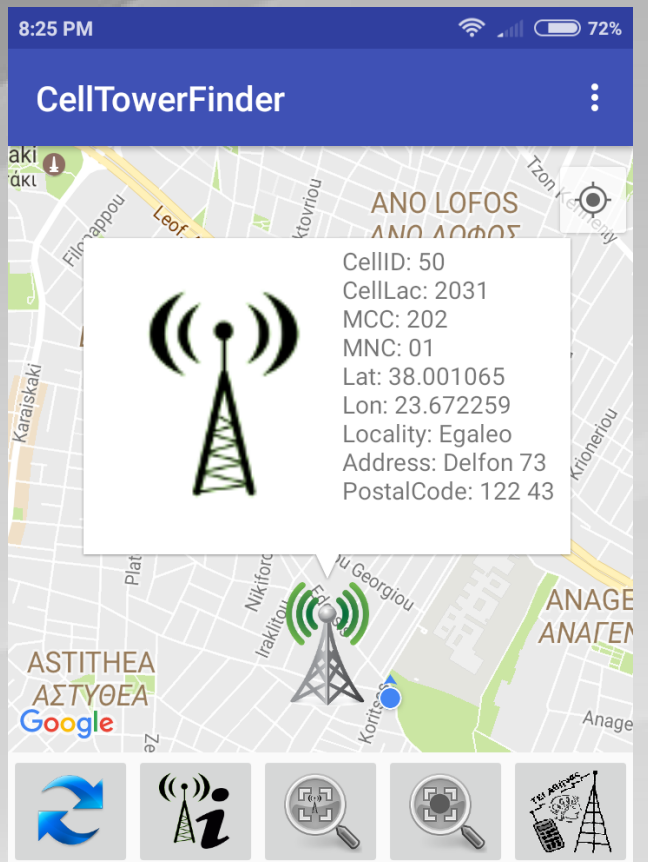


Greece

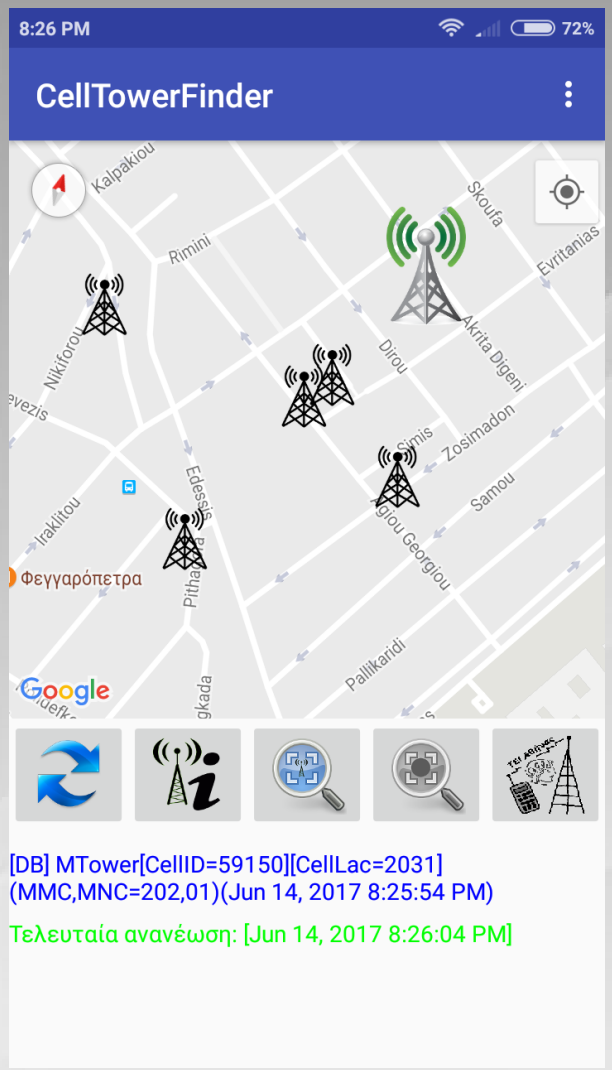
MCC	MNC	Network	Operator or brand name
202	1	Cosmote Mobile Teelecommunications S.A.	Cosmote
202	5	Vodafone - Panafon	Vodafone
202	9	Wind Hellas Telecommunications S.A.	Wind
202	10	Wind Hellas Telecommunications S.A.	Wind

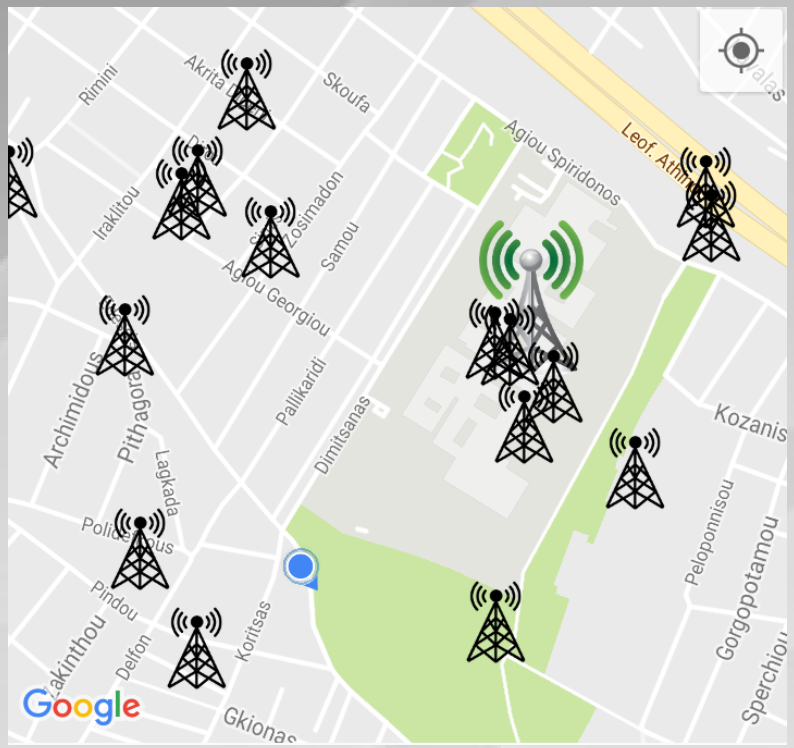
Παρουσίαση εφαρμογής



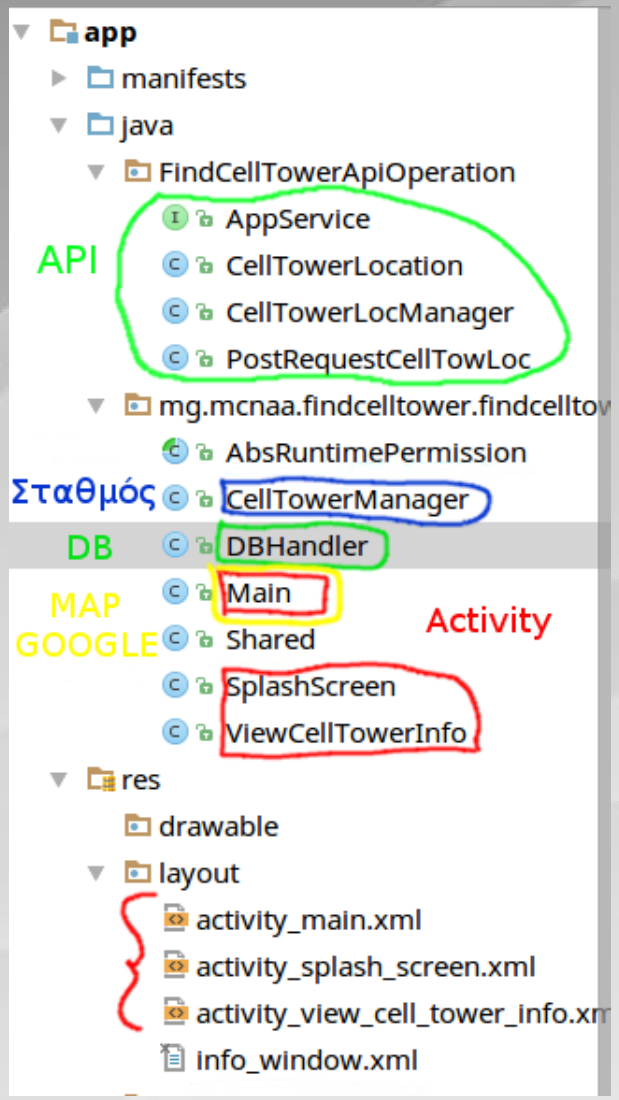


Παρουσίαση εφαρμογής





Τεχνικά χαρακτηριστικά



Τεχνικά χαρακτηριστικά

unwiredlabs LocationAPI

```
@SerializedName("status")
@Expose
private String status;
@SerializedName("message")
@Expose
private String message;
@SerializedName("balance")
@Expose
private String balance;
@SerializedName("lat")
@Expose
private String lat;
@SerializedName("lon")
@Expose
private String lon;
@SerializedName("accuracy")
@Expose
private String accuracy;
@SerializedName("help")
@Expose
private String help = "";
```

```
public class PostRequestCellTowLoc {
    @SerializedName("token")
    @Expose
    private String token;
    @SerializedName("mcc")
    @Expose
    private String mcc;
    @SerializedName("mnc")
    @Expose
    private String mnc;
    @SerializedName("cells")
    @Expose
    private Cells[] cells;
```

```
public class Cells {
    @SerializedName("lac")
    @Expose
    private int lac;
    @SerializedName("cid")
    @Expose
    private int cid;
```

```
public interface AppService {
    @POST("process.php")
    Call<CellTowerLocation> getCellTowerLocation(@Body PostRequestCellTowLoc prctl);
}
```

AppService

PostRequestCellTowLoc

CellTowerLocManager

CellTowerLocation

```
public class CellTowerLocManager {
    public static final String TAG = CellTowerLocManager.class.getSimpleName();
    public static final String BASE_URL = "https://eu1.unwiredlabs.com/v2/";
    public static final String TOKEN = "9f9a8862d44ad0";
```

```
    private CellTowerLocManager() {
        retrofit = new Retrofit.Builder()
                .baseUrl(BASE_URL)
                .addConverterFactory(GsonConverterFactory.create())
                .build();
        appService = retrofit.create(AppService.class);
        cellTowerLocation = new CellTowerLocation();

    }
```

```
public boolean loadCellTowerLocation(String mcc, String mnc, int lac, int cid) {
    boolean returnValue = true;
    PostRequestCellTowLoc prctl = new PostRequestCellTowLoc(TOKEN, mcc, mnc, lac, cid);
    try {
        appService.getCellTowerLocation(prctl).enqueue(new Callback<CellTowerLocation>() {
            @Override
            public void onResponse(Call<CellTowerLocation> call, Response<CellTowerLocation> response) {
                if (response.code() == 200) {
                    CellTowerLocation ctl = response.body();
                    //Log.i(TAG, "Cell Tower Location loaded");
                    getInstance().updateCellTowerLocation(ctl);
                }
            }

            @Override
            public void onFailure(Call<CellTowerLocation> call, Throwable t) {
                //Log.e(TAG, "Error while posting to the api" + t.getMessage());
            }
        });
    } catch (Exception ex) {
        returnValue = false;
    }

    return returnValue;
}
```

Τεχνικά χαρακτηριστικά

Main

```
@Override
public void onLocationChanged(Location location) {
    if (location == null) {
        Shared.showToast(this, "Cant get current location!");
    } else {
        LatLng ll;
        if (zoomGPS) {
            ll = new LatLng(location.getLatitude(), location.getLongitude());
            CameraUpdate update = CameraUpdateFactory.newLatLngZoom(ll, 15);
            mGoogleMap.animateCamera(update);
        }
```

```
private void initMap() {
    MapFragment mapFragment = (MapFragment) getFragmentManager().findFragmentById(R.id.mapFragment);
    mapFragment.getMapAsync(this);
}

@Override
public void onMapReady(GoogleMap googleMap) {
```

```
boolean viewAllCellTowers = false;

private void viewAllCellTowersOperation(boolean viewAll) {
    viewAllCellTowers = viewAll;
    removeAllMarkers();
    if (viewAllCellTowers) {
        bZoomCellTower.setImageResource(R.mipmap.zoomcelltower);
        ArrayList<CellTowerManager> alAllCellsTower = myDB.getAllCellTowers();
        for (CellTowerManager ctm : alAllCellsTower) {
            if (!ctm.getCellTowerAppID().equals(ctiMan.getCellTowerAppID()))
                setMarker(ctm);
            else
                setMarker(ctiMan, true);
        }
    } else {
        bZoomCellTower.setImageResource(R.mipmap.zoomcelltowerdis);
        setMarker(ctiMan, true);
    }

}

boolean zoomGPS = true;

private void zoomOperation(boolean isGPSzoom) {
    zoomGPS = isGPSzoom;
    if (zoomGPS)
        bZoomGPS.setImageResource(R.mipmap.zoomgps);
    else
        bZoomGPS.setImageResource(R.mipmap.zoomgpsdis);
}
```

```
private void setMarker(CellTowerManager ctm) { setMarker(ctm, false); }

private void setMarker(CellTowerManager ctm, boolean isCurrent) {
    removeMarker(ctm);
    MarkerOptions options = new MarkerOptions()
            .title(ctm.getAllInfo())
            .position(new LatLng(ctm.getLat(), ctm.getLon()))
            .icon(BitmapDescriptorFactory.fromResource((isCurrent) ? R.mipmap.celltower2now : R.mipmap.celltower2));

    hmAllMarkers.put(ctm.getCellTowerAppID(), mGoogleMap.addMarker(options));
}
```



Τεχνικά χαρακτηριστικά

ViewCellTowerInfo

```
private void refreshList() {
    try {
        ArrayList<String> alList = CellTowerManager.findAndUpdateLVWithAllCellInfo(telephonyManager);
        ArrayAdapter<String> adapter = new ArrayAdapter<~>(this,
                android.R.layout.simple_list_item_1, alList);
        lvAllInfo.setAdapter(adapter);
    } catch (Exception ex) {
        Shared.fatalError(this, "Error while refreshing the list!");
    }
}
```

CellTowerManager

```
public static ArrayList<String> findAndUpdateLVWithAllCellInfo(TelephonyManager telephonyManager) {
    ArrayList<String> list = new ArrayList<String>();
    if (telephonyManager != null) {
        list.add("All Cell Towers (crowth =" + telephonyManager.getAllCellInfo().size() + ") [Offline Info]");
        list.addAll(getAllCellTowerInfo(telephonyManager));
    }

    return list;
}
```

```
public static ArrayList<String> getAllCellTowerInfo(TelephonyManager tm) {
    List<CellInfo> cellInfos = (List<CellInfo>) tm.getAllCellInfo();

    ArrayList<String> alAllCellTowers = new ArrayList<>();
    for (CellInfo cellInfo : cellInfos)
        alAllCellTowers.add(getCellInfo(cellInfo));

    return alAllCellTowers;
}
```

```
public static String getCellInfo(CellInfo cellInfo) {
    String result = "";
    if (cellInfo instanceof CellInfoLte)
        result = getCellInfoLte((CellInfoLte) cellInfo);
    else if (cellInfo instanceof CellInfoGsm)
        result = getCellInfoGsm((CellInfoGsm) cellInfo);
    else if (cellInfo instanceof CellInfoCdma)
        result = getCellInfoCdma((CellInfoCdma) cellInfo);
    else if (cellInfo instanceof CellInfoWcdma)
        result = getCellInfoWcdma((CellInfoWcdma) cellInfo);

    return result;
}
```

```
public static String getCellInfoLte(CellInfoLte cellInfoLte) {
    StringBuffer buffer = new StringBuffer();
    buffer.append("Type: Lte, Registered =" + ((cellInfoLte.isRegistered()) ? "Yes" : "No") + "\n");
    buffer.append("- - - - - - - - - - - - - - - - - - - - - - - - -\n");
    if (cellInfoLte.isRegistered()) {
        buffer.append("Mcc =" + cellInfoLte.getCellIdentity().getMcc() + "\n");
        buffer.append("Mnc =" + cellInfoLte.getCellIdentity().getMnc() + "\n");
        buffer.append("Ci =" + cellInfoLte.getCellIdentity().getCi() + "\n");
        buffer.append("Tac =" + cellInfoLte.getCellIdentity().getTac() + "\n");
    }
    buffer.append("Pci =" + cellInfoLte.getCellIdentity().getPci() + "\n");
    CellSignalStrengthLte cellSignalStrengthLte = cellInfoLte.getCellSignalStrength();
    buffer.append("AsuLevel =" + cellSignalStrengthLte.getAsuLevel() + "\n");
    buffer.append("Dbm =" + cellSignalStrengthLte.getDbm() + "\n");
    buffer.append("Level =" + cellSignalStrengthLte.getLevel());
    //buffer.append("TimingAdvance ="+cellSignalStrengthLte.getTimingAdvance());

    return buffer.toString();
}
```

Type: Gsm, Registered =No
- -
AsuLevel =0
Dbm =-113
Level =0

Type: Wcdma, Registered =No
- -
AsuLevel =5
Dbm =-103
Level =2

Τεχνικά χαρακτηριστικά

```
@Override
public void onCreate(SQLiteDatabase db) {
    db.execSQL("CREATE TABLE " + TABLE_NAME + "(CellID INTEGER,CellLac INTEGER" +
            ",MCC TEXT, MNC TEXT,Lat REAL,Lon REAL,Info TEXT,Updatedate DATETIME," +
            "PRIMARY KEY (CellID,CellLac,MCC,MNC));");
}
```

DBHandler

```
public class DBHandler extends SQLiteOpenHelper {
    public static final String DATABASE_NAME = "CellTowerPoints.db";
    public static final String TABLE_NAME = "CellTowers";
    private static final SimpleDateFormat DATAFORMAT = new SimpleDate

    public static final String COL_1 = "CellID";
    public static final String COL_2 = "CellLac";
    public static final String COL_3 = "MCC";
    public static final String COL_4 = "MNC";
    public static final String COL_5 = "Lat";
    public static final String COL_6 = "Lon";
    public static final String COL_7 = "Info";
    public static final String COL_8 = "Updatedate";
```

```
public ArrayList<CellTowerManager> getAllCellTowers() {
    SQLiteDatabase db = this.getWritableDatabase();
    Cursor cur = db.rawQuery("SELECT * FROM " + TABLE_NAME, null);
    ArrayList<CellTowerManager> alCellToweres = new ArrayList<>();
    while (cur.moveToNext()) {
        Date date;
        try {
            date = DATAFORMAT.parse(cur.getString(7));
        } catch (ParseException e) {
            date = new Date();
        }
        alCellToweres.add(new CellTowerManager(cur.getInt(0), cur.getInt(1),
                cur.getString(2), cur.getString(3), cur.getDouble(4),
                cur.getDouble(5), cur.getString(6), date));
    }
    cur.close();
    return alCellToweres;
}
```

```
public boolean insertData(CellTowerManager ctm) {
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put(COL_1, ctm.getCellId());
    contentValues.put(COL_2, ctm.getCellLac());
    contentValues.put(COL_3, ctm.getMcc());
    contentValues.put(COL_4, ctm.getMnc());
    contentValues.put(COL_5, ctm.getLat());
    contentValues.put(COL_6, ctm.getLon());
    contentValues.put(COL_7, ctm.getInfo());
    contentValues.put(COL_8, getDateTime());
    long result = db.insert(TABLE_NAME, null, contentValues);

    if (result == -1)
        return false;
    else
        return true;
}
```

Τεχνικά χαρακτηριστικά

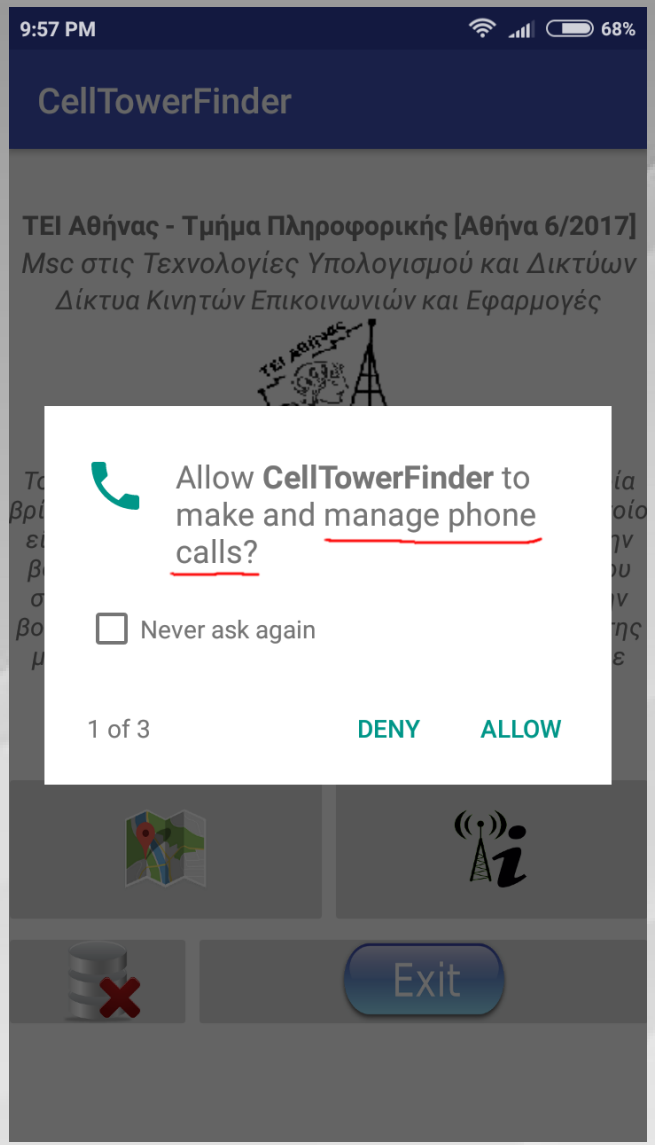
Main

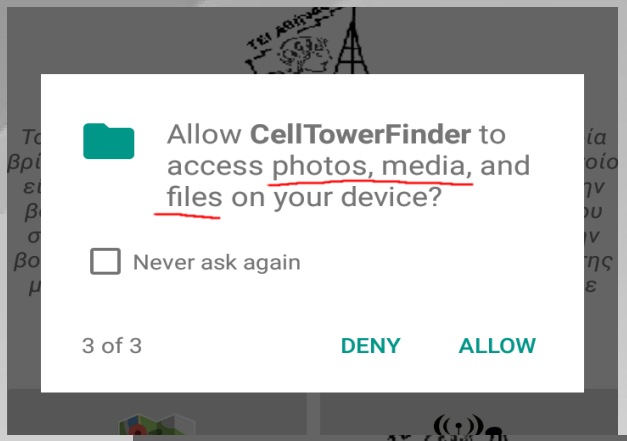
```
private void refresh() {
    changeStatus2("Τελευταία ανανέωση: [" + DateFormat.getDateTimeInstance().format(new Date()) + "]");
    if (!refreshIsBusy) {
        lastCTM = ctiMan.clone();
        int res = ctiMan.reload(telephonyManager) ;
        if (res == 1) {
            changeStatus1("Άλλαξε ο σταθμός βάσης!");
            setRefreshBusy(true);
            CellTowerManager ctm = myDB.getCellTower(ctiMan.getCellId(), ctiMan.getCellLac(), ctiMan.getMcc(), ctiMan.getMnc());

            if (viewAllCellTowers)
                setMarker(lastCTM);
            else
                removeMarker(lastCTM);

            if (ctm != null) {
                changeStatus1("Ο σταθμός βάσης βρέθηκε στην DB!");
                ctiMan = ctm;
                setMarker(ctiMan, true);
                changeStatus1("[DB] MTower[CellID=" + ctiMan.getCellId() + "][CellLac=" + ctiMan.getCellLac() + "] \n" +
                        "(MMC,MNC=" + ctiMan.getMcc() + "," + ctiMan.getMnc() + ")" +
                        "(" + DateFormat.getDateTimeInstance().format(new Date()) + ")");
                setRefreshBusy(false);
            } else {
                changeStatus1("Ο σταθμός βάσης δεν βρέθηκε στην DB\nκαι γίνεται προσπάθεια εύρεσης του από το API...");
                if (CellTowerLocManager.getInstance().loadCellTowerLocation(ctiMan.getMcc(), ctiMan.getMnc(), ctiMan.getCellLac(), ctiMan.getCellId())) {
                    Handler handler = new Handler();
                    handler.postDelayed(() -> {
                            try {
                                double lat = Double.parseDouble(CellTowerLocManager.getInstance().getCellTowerLocation().getLat());
                                double lon = Double.parseDouble(CellTowerLocManager.getInstance().getCellTowerLocation().getLon());

                                ctiMan.setLat(lat);
                                ctiMan.setLon(lon);

                                String info = "";
                                info = "Locality: " + geoCoder.getFromLocation(lat, lon, 1).get(0).getLocality();
                                info += "\nAddress: " + geoCoder.getFromLocation(lat, lon, 1).get(0).getAddressLine(0);
                                info += "\nPostalCode: " + geoCoder.getFromLocation(lat, lon, 1).get(0).getPostalCode();
                                ctiMan.setInfo(info);

                                setMarker(ctiMan, true);
                                changeStatus1("[API] Tower[CellID=" + ctiMan.getCellId() + "][CellLac=" + ctiMan.getCellLac() + "] \n" +
                                        "(MMC,MNC=" + ctiMan.getMcc() + "," + ctiMan.getMnc() + ")" +
                                        "(" + DateFormat.getDateTimeInstance().format(new Date()) + ")");
                                myDB.insertData(ctiMan);
```

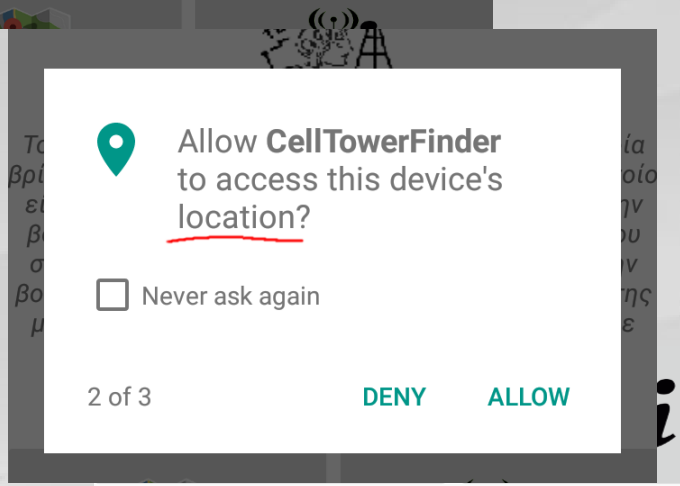
1 2 3 4 5 6

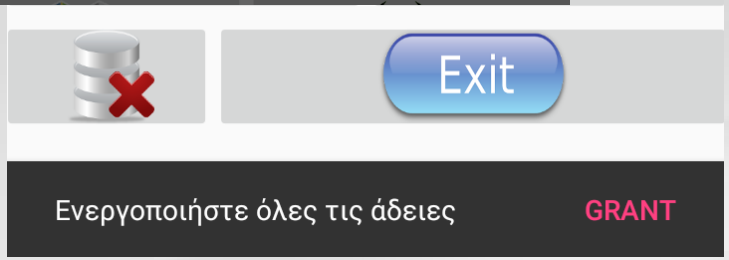


Troubleshooting









Μελλοντικές επεκτάσεις

- Cloud (πχ online DB)
- Καταγραφή διαδρομής
- Καταγραφή απόστασης από κεραία

Συμπεράσματα

- ✓Υπάρχουν **πολλές κεραίες** σε **μικρές αποστάσεις**.
- ✓Ακόμη και όταν είμαστε **σταθεροί**, **αλλάζουμε συχνά σταθμό βάσης**.
- ✓Μάθαμε **πληροφορίες σχετικές** με τους **σταθμούς βάσης** πχ MCC,MNC,Ci,Tac, Pci, Level, Dbm
- ✓Μάθαμε πως μέσα από το **android** (service, api κ.ά), μπορούμε να μάθουμε **πληροφορίες** για **Cell towers**

Ερωτήσεις

Ευχαριστούμε για την προσοχή σας