Esercitazione

Prima parte - Visualizzare una mappa

1) Creare un progetto Android di nome "Runner" che usi le Google Api, dare nome alla activity di default "RunnerActivity" e utilizzare il specificare il package con il nome "pdm.test.mappe"

2) Modificare la RunnerActivity e fargli estendere MapActivity e implementare i metodi richiesti

3) Modificare il layout eliminando la TextView e inserendo una MapView che prenda tutto lo schermo

4) Nel metodo OnCreate prendere il riferimento alla mapview e configurarla abilitando il pan e lo zoom, e la vista con mappe satellitari

```
mapView.setClickable(true);
mapView.setBuiltInZoomControls(true);
mapView.setSatellite(true);
```

NOTA i riferimenti alla MapView ed al MyLocationOverlay, vanno definiti come variabili di istanza

5) Impostare nel manifest

i) i permessi per Internet e per la Localizzazione ACCESS_FINE_LOCATION

ii) l'uso della libreria di google maps nella applicazione

```
<uses-library android:name="com.google.android.maps" />
```

6) Creare un emulatore con target "Google API", con il supporto per il modem GSM ed il supporto per il GPS

7) Testare ed eseguire il programma: provare a muovere la mappa, cambiare zoom, etc.

Seconda parte - Visualizzare la propria posizione

1) Nel metodo onCreate

i) aggiungere un MyLocationOverlay che visualizza la posizione del terminale sulla mappa

```
myLocationOverlay = new MyLocationOverlay(this, mapView);
```

ii) al primo fix della posizione, zommare e animare il centro dello schermo sulla nostra posizione

```
myLocationOverlay.runOnFirstFix(new Runnable() {
    public void run() {
        mapView.getController().animateTo(myLocationOverlay.getMyLocation());
    }
});
```

2) Sovrascrivere i metodi onResume ed onPause inserendo l'attivazione e lo spegnimento delle funzionalità di localizzazione del MyLocationOverlay

myLocationOverlay.enableMyLocation() e myLocationOverlay.disableMyLocation()

3) Testare ed eseguire il programma: provare ad inviare al programma alcune posizioni gps

ES. Stazione Termini lat:41.901222 lon:12.500882

Piazza della Repubblica: lat:41.902622 lon:12.495482

...

NOTA per inserire le coordinate GPS, utilizzare il tool "Emulator Control" e inserire le coordinate con la virgola al posto dei punti

Terza Parte - Visualizzare degli overlay

1) Scaricare la classe RadiusOverlay (RadiusOverlay.java) dal seguente link
"http://ge.tt/8FZlAKD"

2) dichiarare 4 variabili di istanza di tipo RadiusOverlay (la classe appena scaricata)

3) instanziare le 4 variabili nel metodo onCreate utilizzando nei seguenti punti e raggi:
Stazione Termini lat:41.901222 lon:12.500882 r:400m
Piazza della Repubblica: lat:41.902622 lon:12.495482 r:300m
Colosseo: lat:41.890310 lon:12.492410 r:500m
Casa di Romolo e Remo: lat:41.890492 lon:12.484823 r:450m

Usare le costanti della classe android.graphics.Color per specificare il colore nel costruttore. Usare la costante Color.BLUE per questa inizializzazione

ES. di punto

```
GeoPoint termini = new GeoPoint(41902022, 12500882);
overlayTermini = new new RadiusOverlay(termini, 400, Color.BLUE);
mapView.getOverlays().add(overlayTermini);
```

4) Testare il programma muovendo la location GPS per raggiungere le diverse zone

5) Impostate un Alert per l'ingresso/uscita dalla zona termini
i) dichiarare un PendingIntent come variabile di istanza
ii) instanziare nel metodo onResume un PendingIntent da inviare in broadcast con un parametro extra "overlay" che identifica univocamente la zona

```
Intent intentTermini = new Intent("pdm.test.mappe");
intentTermini.putExtra("overlay", 1);
mPendingTermini = PendingIntent.getBroadcast(this, 1, intentTermini, PendingIntent.FLAG_CANCEL_CURRENT);
```

iii) prendere il riferimento al LocationManager ed impostare l'alert di prossimità della stazione termini per lanciare il pending intent

```
LocationManager locationManager = (LocationManager) getSystemService (LOCATION_SERVICE);
locationManager.addProximityAlert(termini.getLatitudeE6() * 0.000001, termini.getLongitudeE6() * 0.000001, 400, -1, mPendingOverlayA);
```

NOTA il raggio di prossimità dovrebbe essere identico a quello utilizzato per il disegno dell'overlay

iv) creare un broadcast receiver come inner class

```
class ProximityBroadcast extends BroadcastReceiver {
    @Override
    public void onReceive(Context arg0, Intent arg1) {
        Log.d(TAG,"Proximity Alert");
        Toast.makeText(getApplicationContext(), "Alert di prossimità",Toast.LENGTH_LONG).show();
    }
}
```

v) dichiarare un ProximityBroadcast come variabile di istanza, crearlo e registrarlo nel metodo onResume()

```
registerReceiver(mProximityBroadcast, new IntentFilter("pdm.test.mappe"));
```

vi) eliminare dalle "registazioni" sia il broadcast receiver che il proximity alert nel metodo onPause()

```
unregisterReceiver(mProximityBroadcast);
locationManager.removeProximityAlert(mPendingTermini);
```

5) Eseguire e testare il programma entrando ed uscendo dalla zona allarmata

6) Aggiungere gli allarmi per le altre zone. Tenere presente le seguenti cose:
    i) per ogni allarme è necessario un PendingIntent
    ii) ogni PendingIntent deve avere un requestCode differente (il secondo parametro del metodo PendingIntent.getBroadcast()
    iii) ogni PendingIntent registrato nell'onResume deve essere eliminato nell'onPause
    iv) ogni Intent, utilizzato da in PendingIntent, deve contenere un valore extra con chiave "overlay" e valore univoco per ciascuna area

7) Eseguire e testare il programma entrando ed uscendo dalle zone allarmate

Quarta Parte - Ingresso e uscita da zone di prossimità

1) Modificare il broadcast receiver in modo da poter distinguere l'ingresso in un area dall'uscita usando il boolean contenuto nell'intent

```
                        boolean stoEntrando = intent.getBooleanExtra
(LocationManager.KEY_PROXIMITY_ENTERING, true);
                        if (stoEntrando){
                             Toast.makeText(getApplicationContext(), "Benvenuto",
Toast.LENGTH_SHORT).show();
                        }else{
                             Toast.makeText(getApplicationContext(), "Arrivederci",
Toast.LENGTH_SHORT).show();
                        }
```

2) Eseguire e testare il programma entrando ed uscendo dalle zone

3) Modificare il Broadcast receiver in modo da rilevare l'area allarmata (fare riferimento alla spiegazione nel punto 6 della terza parte)

```
    int area  = intent.getIntExtra("overlay", -1);
```

4) Modificare il broadcast receiver in modo che:
    i) OGNI volta che l'utente ENTRA in un area, l'area cambia colore (dall'attuale colore al colore verde Color.GREEN)

```
         overlayTermini.setColor(Color.GREEN);
```

    ii) quanto l'utente ESCE dall'area, l'area cambia colore e rimane grigia Color.GRAY

5)Testare che entrando e uscendo da ogni area, le stesse cambino colore da blu a grigio. Rientrando in un area già colorata di grigio, la stessa area deve ridiventare verde.