Scuola universitaria professionale della Svizzera italiana
Dipartimento tecnologie innovative
Istituto sistemi informativi e networking

SUPSI

Localizzazione e Smart Location Library

Sviluppo di Applicazioni Mobile

Vanni Galli, lecturer and researcher SUPSI

Obbiettivi

- Familiarizzare con la gestione dei permessi per accedere alla posizione del device
- Capire come includere una libreria esterna nel progetto
- Imparare a leggere la posizione corrente del device

La posizione del device

- Una delle funzionalità principali (e uniche) di un'applicazione mobile è quella di sapere la posizione geografica corrente del device
- Conoscere la posizione di un device (e quindi di un utente) può aiutare a creare delle applicazioni con un utilizzo più contestuale
- Di norma si richiede all'API l'ultima posizione conosciuta (che di solito coincide con la posizione corrente)
- Le API per la location sono disponibili "nativamente" tramite i servizi *Google Play*

Il Fused Location Provider

- In Android, di norma, si utilizza il *Fused Location Provider* per poter leggere l'ultima posizione conosciuta del device che sta eseguendo l'applicazione

- Il Fused Location Provider fornisce un'API "semplice" (secondo la definizione di Google) per poter avere informazioni sulla posizione corrente del device

- Il Fused Location Provider si occupa di gestire, ad alto livello, le varie "situazioni" in cui si vuole accedere alla posizione (e.g.: situazioni di bassa batteria)

Gestione dei permessi

- Le permission per accedere alla posizione del dispositivo vanno indicate nel file `AndroidManifest.xml`

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.google.android.gms.location.sample.basiclocationsample" >
   <uses-permission
   android:name="android.permission.ACCESS_COARSE_LOCATION"/>
</manifest>
```

- È possibile scegliere tra `ACCESS_COARSE_LOCATION` (la location viene ritornata con una "bassa" precisione) e `ACCESS_FINE_LOCATION` (la location viene ritornata con una precisione più alta)

- Trattandosi di permessi "*dangerous*" servirà anche una richiesta a runtime!

(Non) utilizzo della API

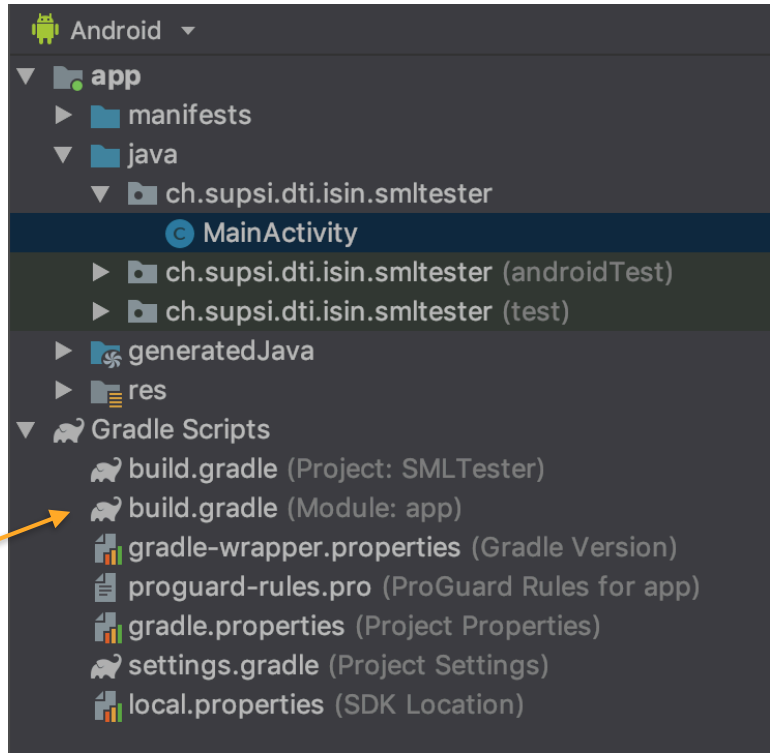
- Malgrado quando indicato nella slide 4, l'utilizzo della API nativa non è "semplice"
- Occorre implementare diversi metodi
- Non è intuitivo capire esattamente il "flusso" delle operazioni
- Ma soprattutto, **le API cambiano spesso**
- Per questo motivo è consigliato utilizzare una libreria esterna che fa un'**ulteriore astrazione** delle API per la localizzazione

La Smart Location Library

- La Smart Location Library è un progetto che intende facilitare l'utilizzo dei location providers in Android
    - È sviluppata da terzi
    - Ha licenza MIT
    - Per poterla utilizzare va dapprima inclusa nella propria applicazione
- È reperibile presso https://github.com/mrmans0n/smart-location-lib
- Gli ultimi aggiornamenti alla libreria risalgono a parecchi anni fa, si tratta dunque di una libreria "vecchia", che però continua a funzionare in modo egregio

Aggiunta di una libreria esterna ad un'applicazione

- Le librerie esterne vanno aggiunte direttamente nel file `build.gradle` del *Module*:



attenzione a selezionare
il file giusto!

Aggiunta della Smart Location Library al progetto

- La Smart Location Library viene aggiunta al file `build.gradle` nel seguente modo:

```
dependencies {
  implementation fileTree(dir: 'libs', include: ['*.jar'])
   implementation 'com.android.support:appcompat-v7:28.0.0'
   implementation 'com.android.support.constraint:constraint-layout:1.1.3'
   testImplementation 'junit:junit:4.12'
   androidTestImplementation 'com.android.support.test:runner:1.0.2'
   androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2'
  implement 'io.nlopez.smartlocation:library:3.3.3'
}
```

Utilizzo della Smart Location Library

- La Smart Location Library è estremamente semplice da usare
- Nel repository si trovano già parecchi esempi
- La posizione può essere letta con il seguente metodo:

```
private void startLocationListener() {
  LocationParams.Builder builder = new LocationParams.Builder()
    .setAccuracy(LocationAccuracy.HIGH)
    .setDistance(0)
    .setInterval(5000); // 5 sec

  SmartLocation.with(this).location().continuous().config(builder.build())
    .start(new OnLocationUpdatedListener() {
       @Override
       public void onLocationUpdated(Location location) {
         Log.i(TAG, "Location" + location);
  }});
}
```

Il Builder dei parametri

- Come la libreria "nativa" di google, anche la Smart Location Library fa utilizzo di un builder per definire i parametri

- I parametri sono:
  - L'accuratezza della posizione (`Accuracy`)
  - L'intervallo con cui richiedere la posizione (`Interval`)
  - La distanza entro la quale una posizione viene considerata uguale (`Distance`)

- La libreria mette già a disposizione dei presets, come ad esempio:

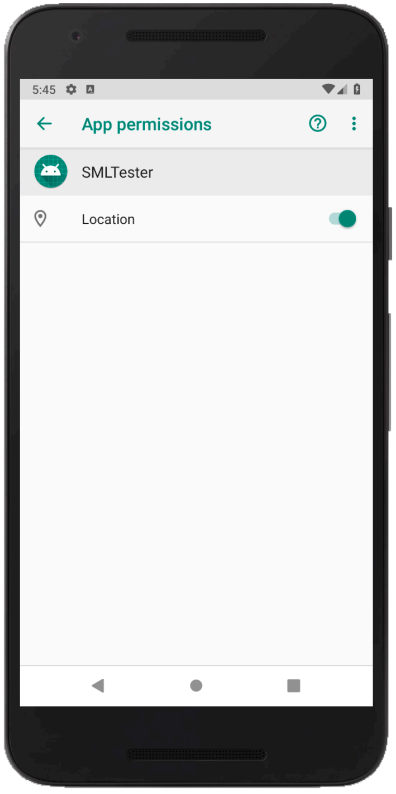
```
public static final LocationParams NAVIGATION = new
Builder().setAccuracy(LocationAccuracy.HIGH).setDistance(0).setInterval(500).build();
```

L'autorizzazione a leggere la posizione

- Indicare le permission per accedere alla posizione all'interno del file `AndroidManifest.xml` non basta

- A partire da Android 6.0, per alcune permission (quelle marcate come `dangerous`) è richiesta una conferma a runtime

- Se non viene implementato nessun meccanismo di richiesta delle permission, l'utente deve **abilitare manualmente** una determinata permission nei settings del device



Controllo dei permessi

- Prima di utilizzare funzionalità come la lettura della posizione del dispositivo, è dunque buona prassi controllare di avere i permessi per farlo

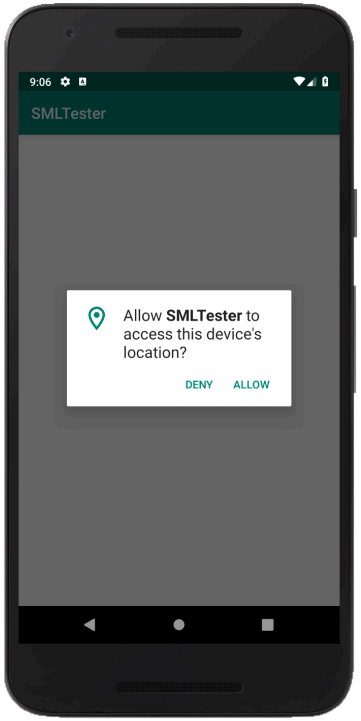
- È possibile implementare logiche diverse nel caso si siano già ottenuti i permessi in precedenza o meno:

```
if (ContextCompat.checkSelfPermission(MainActivity.this,
Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
  Log.i(TAG, "Permission not granted");
  // richiedo i permessi all'utente
} else {
  Log.i(TAG, "Permission granted");
  // ho già i permessi: leggo la posizione del device
}
```

Richiesta di un permesso

- Un permesso può essere richiesto a runtime all'utente:

```
if (ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION)
!= PackageManager.PERMISSION_GRANTED) {
  ActivityCompat.requestPermissions(this, new String[]{Manifest.permission.ACCESS_FINE_LOCATION}, REQ_CODE);
} else {
  // ho già i permessi
}
```



tipo di permesso

request code che verrà tornato in risposta

Gestione della risposta ad una richiesta

- La risposta data dall'utente alla richiesta di un permesso viene intercettata nel metodo `onRequestPermissionsResult`

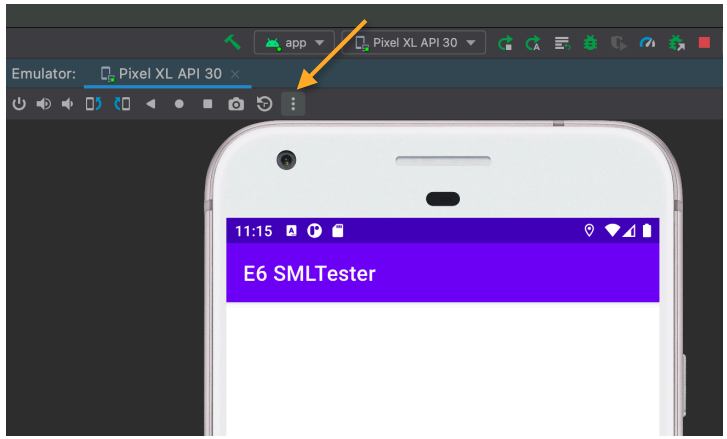
- Il request code specificato in precedenza serve per discriminare tra le varie richieste

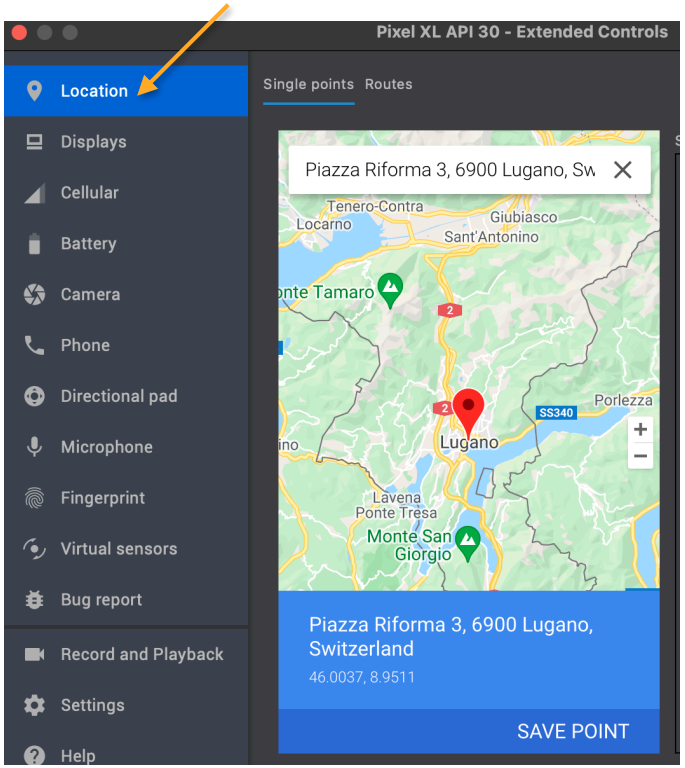
```
@Override
public void onRequestPermissionsResult(int requestCode, String permissions[], int[] grantResults) {
  switch (requestCode) {
    case REQ_CODE: {
      if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
        // ho ottenuto i permessi
      }
      return;
    }
  }
}
```

Emulare la posizione

- È possibile emulare la posizione di un dispositivo direttamente dall'emulatore Android

- Una volta fatto partire l'emulatore, basta selezionare l'ultima voce del menu verticale e scegliere "Location":





Test della Smart Location Library

- Il progetto *SMLTester* su iCorsi permette di testare la Smart Location Library
- Una volta emulata la posizione del dispositivo, è possibile verificarla direttamente a console