Train Radar
Documentazione tecnica

Bortolin Alessandro, 5BIA

8 febbraio 2020

Sommario

Train Radar è l'applicazione che ti permette di seguire in tempo reale i treni in tutta Italia! Questa applicazione si ispira a *flighradar24* [7], un'applicazione che mostra la posizione degli aerei in tutto il mondo in diretta, tuttavia non esiste alcuna applicazione simile sul Play Store per i treni. Grazie a *Train Radar* puoi monitorare i treni di qualsiasi regione, ordinarli in base alla propria distanza e monitorare i ritardi di ogni treno!

1 Dettagli tecnici

1.1 Activity

L'applicazione possiede tre activity con cui l'utente può interagire:

`RadarActivity` è l'activity principale, mostra all'utente la mappa con la posizione dei treni e permette di cercare un treno specifico.

`TrainActivity` mostra le informazioni principali del treno selezionato con la lista delle fermate e la mappa del tragitto.

`NearTrainsActivity` mostra una lista con tutti i treni ordinati per distanza.

1.2 Fragment

`TrainDetailFragment` viene mostrato all'interno della `RadarActivity` quando l'utente interagisce con un treno, mostra le informazioni principali del treno.

`TrainRouteFragment` si trova dentro il `ViewPager` all'interno della `TrainActivity`, mostra le informazioni principali di un treno ed il suo tragitto.

`TrainMapFragment`, come il `TrainRouteFragment`, si trova dentro il `ViewPager` all'interno della `TrainActivity` attraverso un `ViewPager`, mostra la mappa con il tragitto del treno.

1.3 MapView

L'applicazione possiede una `MapView` [3] per mostrare la mappa dei treni limitata al territorio italiano.

1.4 Intent

L'applicazione possiede un `Intent` per passare dalla `RadarActivity` alla `NearTrainsActivity` in cui viene passata la posizione. Inoltre sia la `RadarActivity` che la `NearTrainsActivity` possiedono un `Intent` per avviare la `TrainActivity` in cui vengono passate le informazioni relative al treno selezionato. Agli `Intent`, se possibile, vengono passate informazioni ausiliarie in modo da animare la transizione tra le activity al meglio [4].

1.5 Handler

La posizione di ogni treno mostrato nell'activity principale viene aggiornata in tempo reale. Questa operazione viene eseguita mediante un `Handler` in modo da programmare la sua esecuzione ad un intervallo regolare di un secondo.

1.6 AsyncTask

Calcolare la posizione dei treni è un'operazione relativamente pesante e richiede approssimativamente 100ms. Per dare un'interfaccia fluida all'utente tale operazione viene eseguita in background mediante un `AsyncTask`.

1.7 Internet

L'applicazione necessita di una connessione ad internet in modo da visualizzare la mappa con i treni e aggiornare in tempo reale il ritardo dei treni [1].

1.8 Geolocalizzazione

L'utente può anche acconsentire all'applicazione di usare i dati relativi alla propria posizione geografica, in tal modo l'utente può visualizzare i treni nelle proprie vicinanze riordinati per distanza.

1.9 Supporto per schermi orizzontali

L'applicazione supporta sia lo schermo in modalità verticale sia in modalità orizzontali [2]. Le activity sono state ottimizzate in modo da supportare al meglio l'orientamento dello schermo.

Nell'activity `RadarActivity` la descrizione del treno selezionato viene mostrata in basso quando lo schermo è verticale mentre viene mostrata di lato quando lo schermo è orizzontale.

Nell'activity `TrainActivity` viene mostrato un layout scrollabile con due pagine in modalità verticale, mentre in modalità orizzontale vengono mostrate entrambe le pagine (figure 1 e 2).

Nell'activity `NearTrainsActivity` in modalità orizzontale la lista dei treni viene visualizzata con due colonne anziché una sola.

1.10 Librerie di terze parti

La libreria *Gson* [6] è stata utilizzata per serializzare e deserializzare i dati in formato json.

La libreria *Volley* [5] è stata utilizzata per eseguire le richieste HTTP delle API di *Trenitalia*.

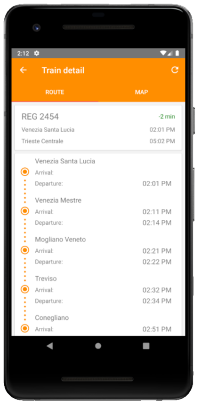

Figura 1: Schermo in modalità verticale

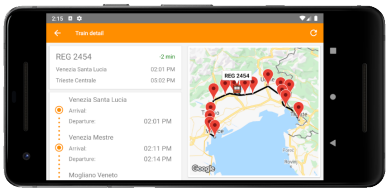

Figura 2: Schermo in modalità orizzontale

2 Caratteristiche principali

La caratteristica principale dell'applicazione è la possibilità di monitorare in tempo reale la posizione dei treni in tutta Italia, questo aspetto contraddistingue *Train Radar* da tutte le altre applicazioni presenti nel Play Store. L'applicazione inoltre fornisce i ritardi di ogni treno in modo da garantire affidabilità agli utenti. Un'altra caratteristica importante è la versatilità poiché l'applicazione mira ad un pubblico vasto e non necessita di particolari conoscenza per utilizzarla, inoltre per garantire al meglio la versatilità l'applicazione supporta in modo completo sia la visualizzazione in verticale che in orizzontale. *Train Radar* necessita di una connessione ad internet per poter funzionare tuttavia l'applicazione dispone di una cache per ridurre al minimo tale consumo e poter utilizzarla in qualsiasi situazione.

3 Struttura dell'applicazione

L'applicazione è molto semplice ed intuitiva, nell'activity principale è presente una mappa con cui l'utente può interagire, muovendosi e visualizzando i treni nella zona selezionata, clickando sopra di un treno l'utente può visualizzare le informazioni relative a tale treno. Nella barra superiore dell'activity è presente un menù con cui l'utente può spostarsi alle altre activity. Se l'utente acconsente a fornire le informazioni relative alla propria posizione, nella mappa verrà mostrato un marcatore a segnalarla e clickando tale marcatore si aprirà un'activity mostrante la lista di tutti i treni presenti ordinati per distanza assoluta. Clickando a sua volta un treno della lista si aprirà un'ulteriore activity contenente tutte le informazione del treno ed una mappa con il suo tragitto. Quest'ultima activity è anche accessibile dall'activity principale interagendo con un determinato treno.

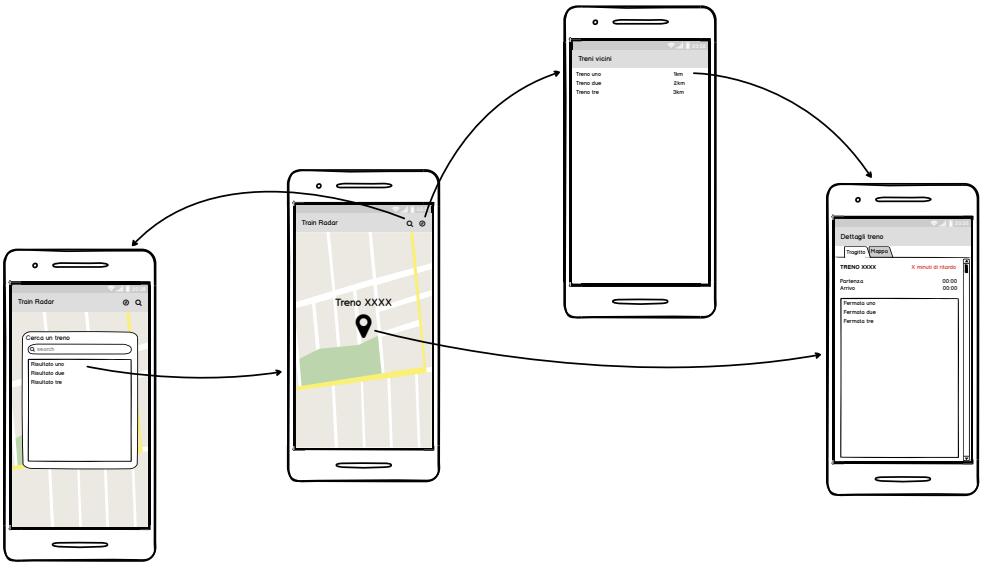

Figura 3: Wireframe dell'applicazione

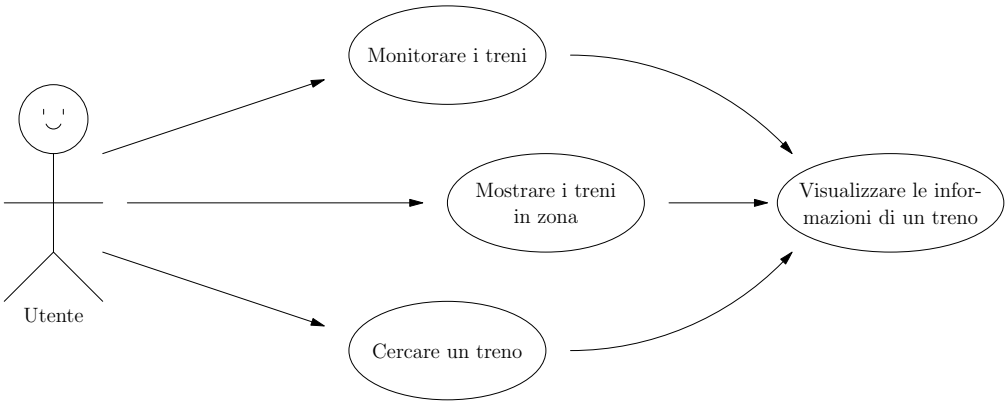

Figura 4: Diagramma d'uso dell'applicazione

4 Frammenti di codice

Il cuore dell'applicazione sono le classi base che rappresentano i treni e le stazioni, esse contengono le informazioni principali di ogni treno e stazione.

```
public class Train {
    private int id;
    private String name;
    private String idDeparture;
    private String idArrival;
    private LocalTime departure;
    private LocalTime arrival;
    private ArrayList<Stop> stops;
    ...
}
```

```
public class Station {
    private String id;
    private String name;
    private LatLng position;
    private ArrayList<String> links;
    ...
}
```

Ad ogni treno mostrato nella mappa viene associato un `Marker` in cui viene sostituita l'icona di default con una rappresentante un treno.

```
Marker marker = googleMap.addMarker(new MarkerOptions()
    .position(pos)
    .title(train.getName())
    .icon(BitmapDescriptorFactory.fromBitmap(TrainManager.getTrainIcon())));
marker.setTag(train);
```

Per aggiornare la posizione di treno viene usato un `Handler` che ricalcola la posizione del treno ogni secondo.

```
public class Scheduler {
    private static final Handler handler = new Handler();

    public static void scheduleTracker(Marker marker, long delay) {
        Train train = (Train) marker.getTag();
        handler.post(new Runnable() {
            @Override
            public void run() {
                LatLng pos = train.getPosition(TimeManager.now());
                marker.setPosition(pos);
                handler.postDelayed(this, delay);
            }
        });
    }
}
```

Per calcolare la posizione di un treno tra due stazioni si usa l'algoritmo di Dijkstra in modo da trovare il percorso atteso meglio approssimato.

```
Set<Station> visited = new HashSet<>();
Map<Station, Double> dist = new HashMap<>();
Map<Station, Station> prev = new HashMap<>();
Queue<Pair<Double, Station>> queue = new PriorityQueue<>((a, b) -> Double.compare(a.first, b.first));
queue.add(new Pair<>(0d, first));
dist.put(first, 0d);

while (!queue.isEmpty()) {
    Pair<Double, Station> curr = queue.poll();
    if (visited.contains(curr.second)) continue;
    visited.add(curr.second);
```

```
    if (curr.second == last) break;
    for (String id : curr.second.getLinks()) {
        Station nextSt = StationManager.getStation(id);
        double nextDist = curr.first + curr.second.getLocation().distanceTo(nextSt.getLocation());
        Double currNextDist = dist.get(nextSt);
        if (currNextDist == null || nextDist < currNextDist) {
            dist.put(nextSt, nextDist);
            prev.put(nextSt, curr.second);
            queue.add(new Pair<>(nextDist, nextSt));
        }
    }
}
```

5 Sviluppo

Target API level:	29
Minimum API level:	26
IDE:	Android Studio

5.1 Difficoltà riscontrate

Calcolo della posizione dei treni *Trenitalia* non fornisce direttamente la posizione dei treni, perciò è stato richiesto un *workaround* per ottenerla. Per prima cosa, grazie alle API di *Trenitalia* [1] si possono ricavare le coordinate geografiche della maggior parte delle stazione, successivamente si può calcolare un'approssimazione della posizione calcolando la posizione attesa in linea d'aria tra l'ultima stazione visitata e la successiva stazione. Tuttavia questa tecnica fallisca con treni a lunga tratta (ad esempio Milano-Roma senza fermate intermedie), per migliorare ulteriormente la posizione si può rappresentare le ferrovie italiane come un grafo ed utilizzare l'algoritmo di Dijkstra [8] per ottenere un percorso più preciso.

Gestione dei thread La posizione dei treni viene calcolata in background in modo da lasciare fluida l'esperienza dell'utente, tuttavia ciò ha richiesto un ulteriore sforzo per sincronizzare i thread. Il principale problema avveniva quando si cambiava activity mentre stava avvenendo un aggiornamento della posizione causando un crash dell'applicazione. Il problema è stato risolto mediante i monitor del Java.

Transizioni delle activity Nelle transizioni tra due activity vengono visualizzate delle animazioni [4] dove alcuni elementi comuni alle due activity vengono mantenuti, tuttavia ciò non funzionava nella activity `TrainActivity` poiché utilizzando un `ViewPager` il contenuto dell'activity veniva visualizzato attraverso un `Fragment` che veniva processato leggermente più tardi rispetto alla creazione dell'activity. Il problema è stato risolto utilizzando `postponeEnterTransition()` e `startPostponedEnterTransition()`.

5.2 Bug noti

Tratte non aggiornate *Train Radar* salva tutte le tratte in un database locale, tuttavia dato che *Trenitalia* cambia frequentemente gli orari delle proprie tratte, può capitare che un treno in realtà cancellato venga comunque mostrato e viceversa un treno in programma non venga mostrato. Purtroppo l'unico modo per correggere questo bug è aggiornare quotidianamente tutte le tratte che però è un'operazione lunga e pesante e non fattibile in pratica.

Ritardo dei treni Per rendere il consumo di dati più leggero, *Train Radar* calcola il ritardo solo dei treno mostrati nella mappa. Questo implica che se un treno è già arrivato in programma, ma in realtà ha accumulato un ritardo e perciò è ancora in viaggio, *Train Radar* supponendo che sia già arrivato non calcola il ritardo di tale treno e perciò non lo mostra nella mappa.

5.3 Sviluppi futuri

Per adesso *Train Radar* mostra la posizione dei treni solo di *Trenitalia* e di *Trenord*, tuttavia in Italia ci sono decine di altre imprese ferroviare come ad esempio *Italo*. L'applicazione può essere migliorata aggiungendo il supporto a queste società ferroviarie.

5.4 Autovalutazione

Nel complesso darei un voto di 4.5 su 5 all'applicazione poiché è molto utile, ben strutturata, semplice e si rivolge verso un pubblico di molto vasto. Chiunque con la passione dei treni apprezzerebbe questa applicazione.

Riferimenti bibliografici

[1] bluviolin. *API del sistema Viaggiatreno*. URL: `https://github.com/bluviolin/TrainMonitor/wiki/API-del-sistema-Viaggiatreno`.

[2] Android Developers Documentation. *Device compatibility overview*. URL: `https://developer.android.com/guide/practices/compatibility`.

[3] Android Developers Documentation. *Maps SDK for Android*. URL: `https://developers.google.com/maps/documentation/android-sdk/start`.

[4] Android Developers Documentation. *Start an activity using an animation*. URL: `https://developer.android.com/training/transitions/start-activity`.

[5] Android Developers Documentation. *Volley overview*. URL: `https://developer.android.com/training/volley`.

[6] Google Inc. *Gson*. URL: `https://github.com/google/gson`.

[7] Svenska Resenätverket. *flighradar24*. URL: `https://www.flightradar24.com/`.

[8] Wikipedia. *Dijkstra's algorithm*. URL: `https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm`.