ThingSpeak4Android

Natale Vadalà
Matr.0000691364

Introduzione

L' applicazione, disponibile per dispositivi Android, nasce con l'obiettivo di monitare i sensori di uno smartphone, raccogliere questi risultati su un server e riuscire a visualizzarli in un secondo momento direttamente sul cellulare.

Per ottenere ciò (e al meglio), l'utente dovrà consentire all'app di ottenere vari permessi:

- android.permission.CHANGE_WIFI_STATE
- android.permission.ACCESS_WIFI_STATE
- android.permission.INTERNET
- android.permission.ACCESS_NETWORK_STATE
- android.permission.ACCESS_COARSE_LOCATION
- android.permission.CHANGE_NETWORK_STATE
- android.permission.BODY_SENSORS
- android.permission.RECEIVE_BOOT_COMPLETED
- android.permission.ACCESS_COARSE_LOCATION
- android.permission.ACCESS_FINE_LOCATION
- android.permission.ACCESS_LOCATION_EXTRA_COMMANDS
- com.android.alarm.permission.SET_ALARM

Le features dell'applicazione, fondamentalmente, sono tre (come descritto dalla specifica):

1) Sensor Configuration
2) Send Data
3) Get Statistics

Rispettivamente consistono in:

(1) Visualizzare tutti i sensori disponibili (inclusi dati relativi a connessione Wifi e Mobile) e poter selezionare alcuni sensori per iniziare l'attività di monitoring e reporting.

(2) Iniziare l'attività di monitoring e di comunicazione con il server, attraverso un servizio che risvegli l'attività principale anche dopo l'uscita dall'applicazione, e che parta al boot del cellulare. Possibilità di modificare la frequenza di reporting dei sensori, e di scegliere se usare la rete Wifi, la connessione dati (o qualsiasi connessione) per comunicare i propri risultati al server.

(3) Scaricare i grafici del proprio reporting dal server, selezionando i dati dell'ultimo giorno/settimana/mese.

Progettazione

Per progettare l'app, è stato necessario appoggiarsi ad alcuni servizi esterni per mantenere i dati e disegnare i grafici.

"*ThingSpeak*"[1] è il nome del servizio utilizzato per conservare i dati: gratuito, ben dotato di metodi per la creazione, l'aggiornamento e l'eliminazione di canali direttamente con richieste Get/Post/Delete, semplice da utilizzare e da interrogare (poiché i dati vengono mantenuti sul server in JSON). Dà la possibilità, inoltre, di accedere ai propri grafici dal sito web e esportare grafici ben più complessi.

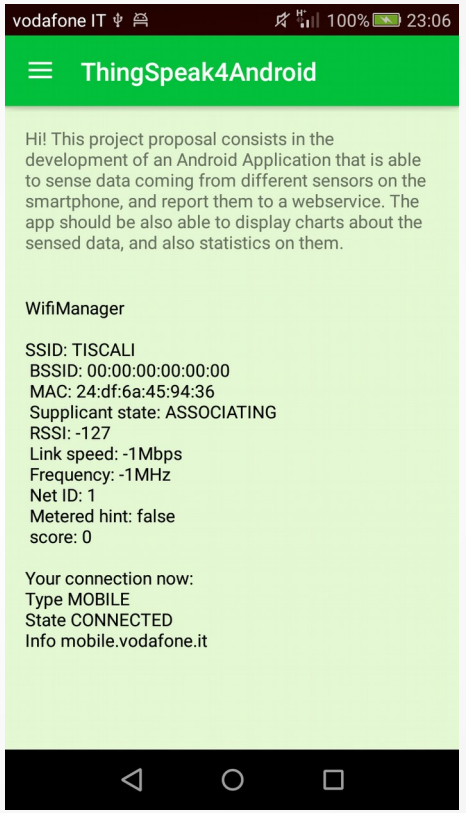
"*HelloChart*"[2], libreria per Android, che permette di rappresentare grafici su app Android in modo semplice e, devo dire, molto pulito.

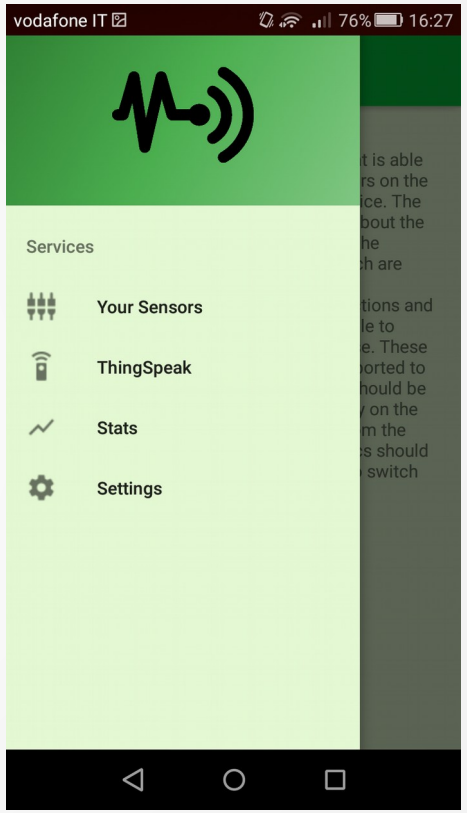
Ambiente di sviluppo: è stato utilizzato l'IDE "*Android Studio*"[3] per la progettazione con Android 5.0 e Api level 21, diversi dispositivi Android collegati alla console per la fase di testing.

MainActivity

MainActivity: è una NavigationDrawerActivity, fornisce l'accesso a tutte le altre Activities.

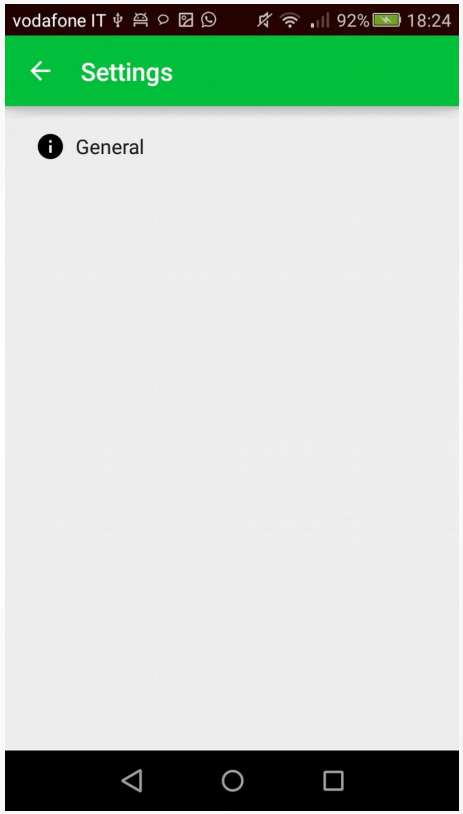
Contiene una piccola descrizione del progetto, le informazioni inerenti al nostro WifiManager e alle nostre connessioni (quale tipo di connessione stiamo usando al momento).

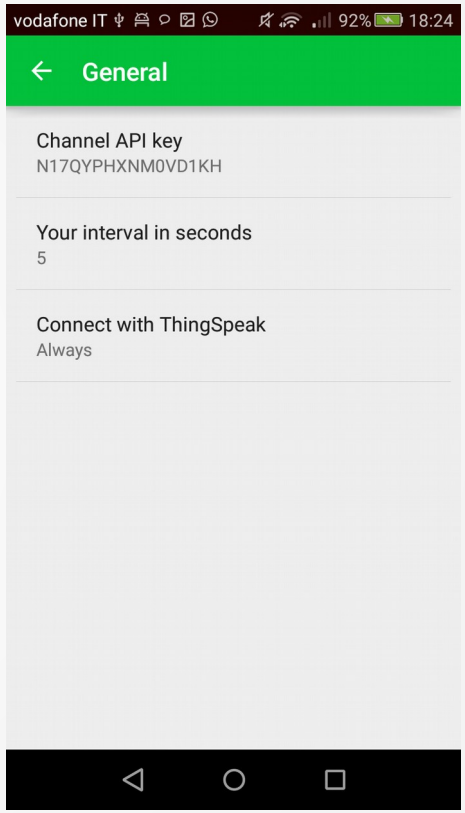




SettingsActivity

SettingsActivity, è una PreferenceActivity.
Permette di
1) cambiare account (cambiando la Master API-Key di Thingspeak);
2) cambiare la frequenza di reporting;
3) selezionare che tipo di connessione utilizzare.

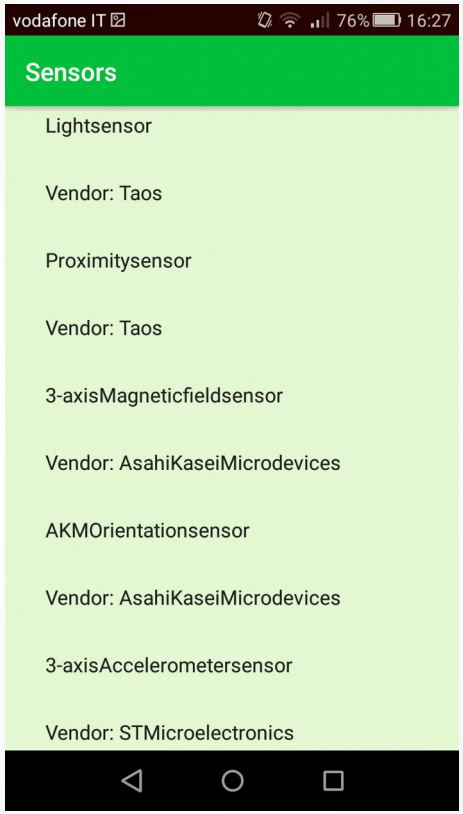


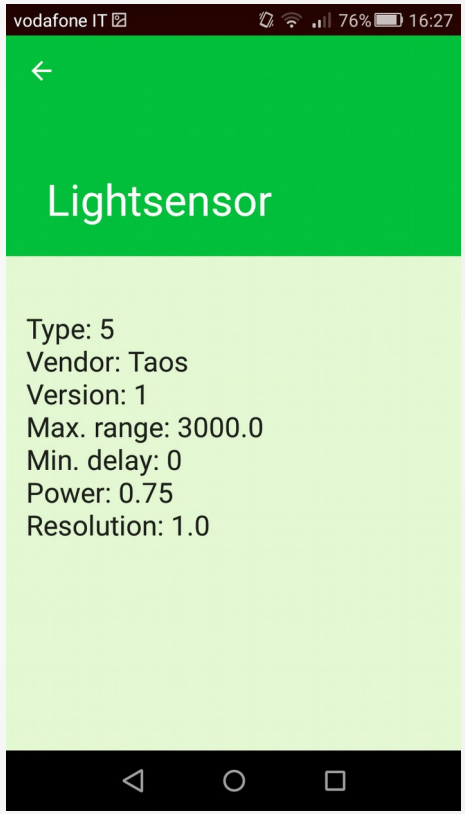


SensorListActivity

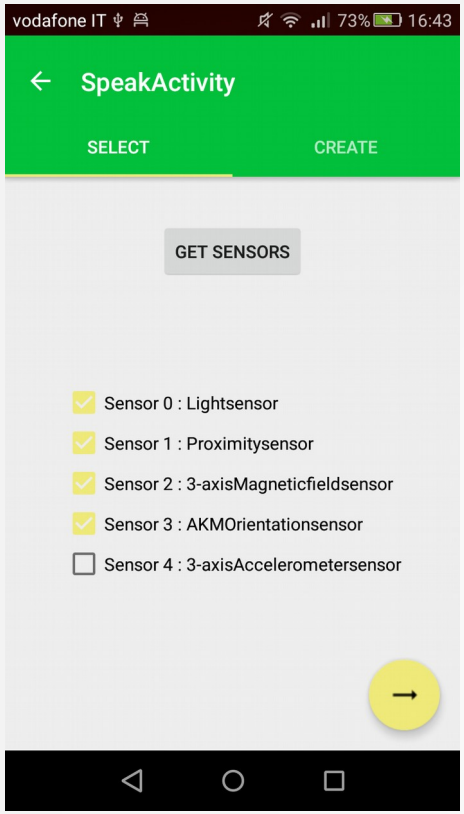
SensorListActivity: è una MasterDetailActivity, con la possibilità (su tablet) di visualizzare tutto su una sola tab, infatti è costituita dalla SensorDetailActivity e dal SensorDetailFragment.

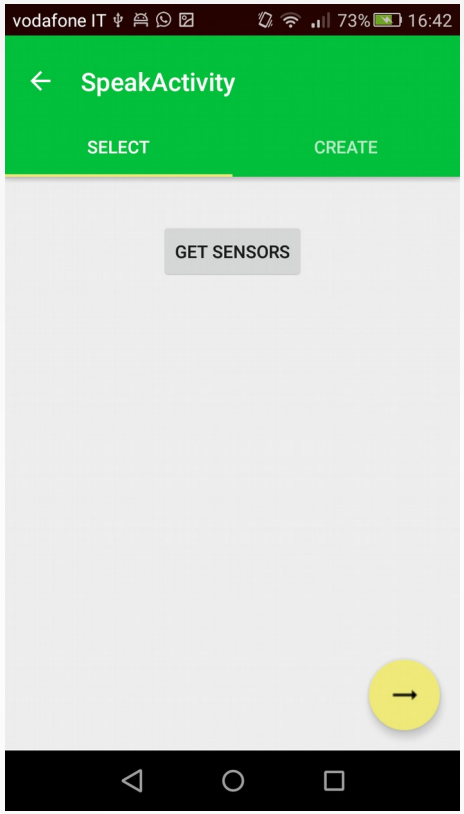
All'avvio inizializza tutti i sensori disponibili e permette di visualizzarli in una lista espandibile, che mostra i dettagli di ciascun sensore rilevato.

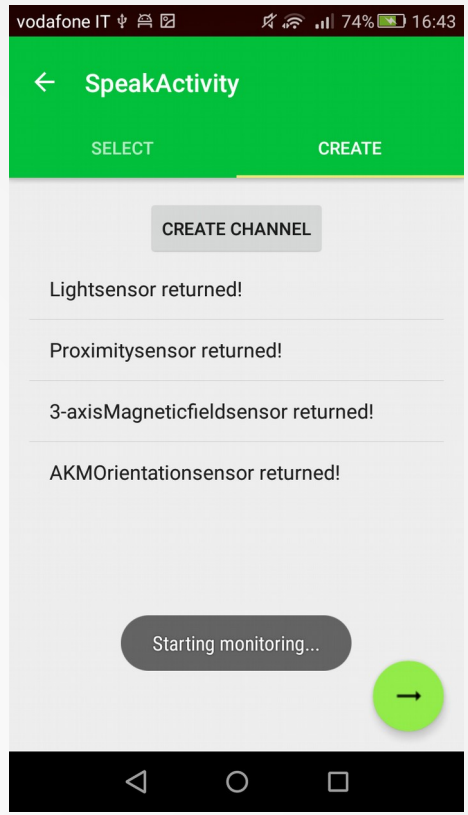




SpeakActivity



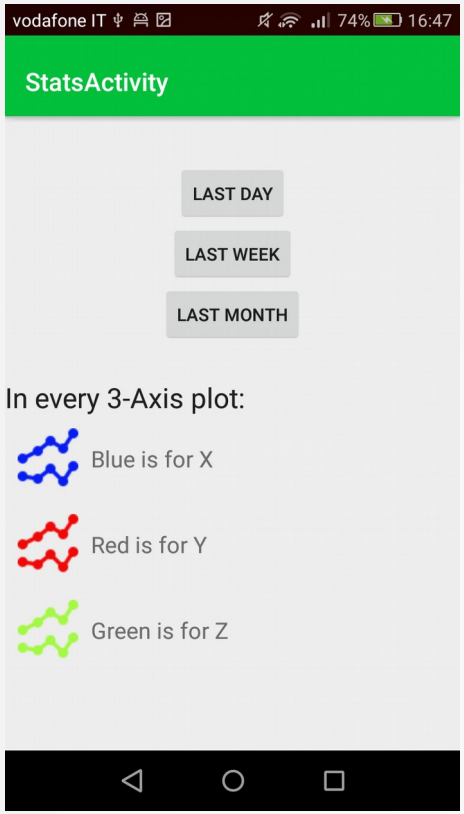


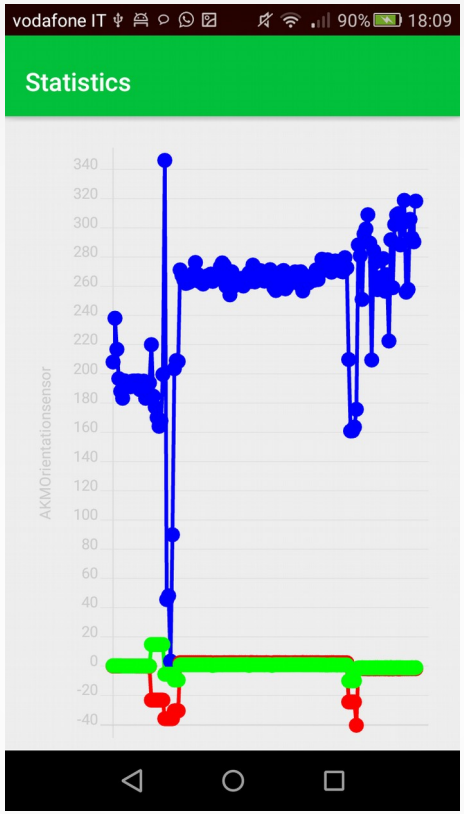


SpeakActivity è una TabbedActivity, ed è quella che ci permette di fare un mini “setup” della nostra applicazione, rilevando i sensori, dandoci la possibilità di scegliere quale tracciare e di creare i canali per Thingspeak.

Alla fine di tutti gli step, se tutto è andato a buon fine (fondamentalmente, se c'è abbastanza linea per non fare andare in timeout le post), sarà stato creato un canale per ogni sensore.
Il FloatingButton permette di iniziare l'attività principale: quella di monitoring e reporting al server.

StatsActivities



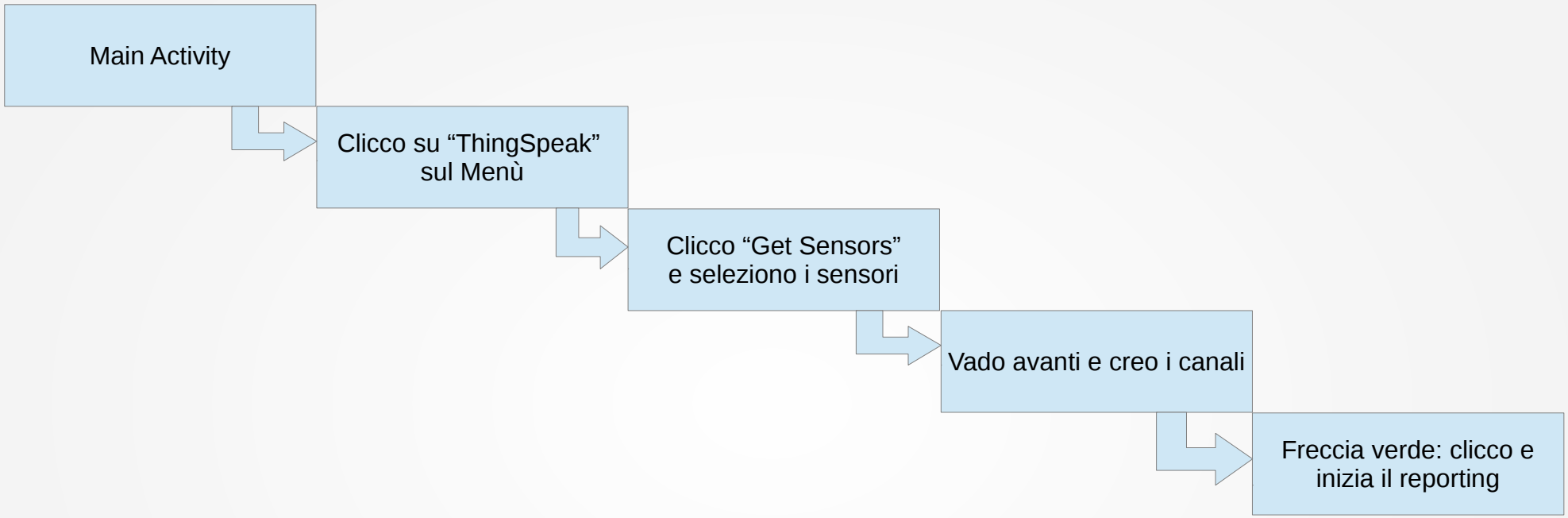


StatsActivity e **Stats2Activity** sono, rispettivamente, quella che chiede i feed dei canali al server e disegna il grafico, e quella che li rappresenta su una TabbedActivity.

MainService

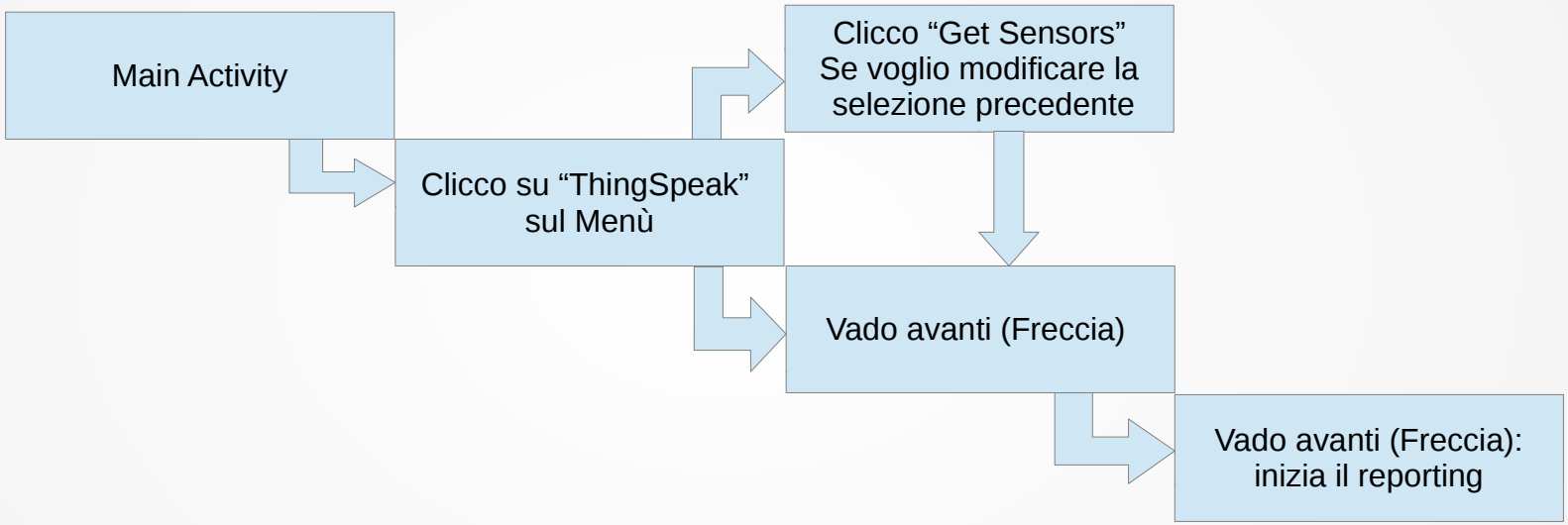
MainService è il nucleo dell'applicazione, la classe che implementa un SensorEventListener per ottenere i valori dei sensori e genera gli update da mandare al server (con chiamate asincrone ottenute attraverso la classe *AsyncTask*).
Il service è in background, ha la capacità di rimanere vivo anche dopo la distruzione di tutte le Activities grazie al **MyBroadcastReceiver** che lo risveglia, anche dopo il reboot dello smartphone.

Primo avvio



NOTA: Per evitare all'utente problemi in futuro, nel momento in cui avvenisse una richiesta di monitoring di un sensore che non è stato mai registrato sul nostro account ThingSpeak, alla prima CreateChannel dopo l'installazione vengono creati (preventivamente) comunque i canali per ogni sensore sullo smartphone, anche se l'attività di monitoring su quel sensore non verrà mai eseguita (perché non selezionato dall'utente).
Dopo la prima, ogni CreateChannel, se eseguita per un sensore già registrato, sovrascrive le relative api-keys precedenti (Accedendo all'account ThingSpeak troveremo comunque tutti i grafici creati).

Reporting


Main Activity
Clicco su “ThingSpeak” sul Menù
Clicco “Get Sensors” Se voglio modificare la selezione precedente
Vado avanti (Freccia)
Vado avanti (Freccia): inizia il reporting

Statistiche

Service

Le problematiche incontrate sono state di diverso tipo:
a) Scrittura del Service, la parte riguardante i sensori e la frequenza di aggiornamento, risolto implementando un SensorEventListener che, al momento del cambio di valore di un sensore (la prima volta), questo viene “unregistrato” dal Listener, per essere riregistrato al tempo T richiesto dall'utente attraverso un Handler ed una richiesta PostDelayed.

b) Scrittura del Service, la parte riguardante l'avvio al boot e l'avvio alla distruzione dell'app, risolto banalmente cambiando telefono (Huawey P8 Light) e leggendo su Internet: installando app da debug usb, Huawey non dà la possibilità di acconsentire all'esecuzione in background (bisogna buildare l'APK e installarlo da telefono)

Altre problematiche

c) Scrittura della CreateChannel, poiché teoricamente dovrebbe essere una chiamata sincrona (fino a che non ho i canali non posso scriverci i miei dati), risolto simulando il sincronismo attraverso chiamate asincrone.

d) Gestione Preference “Only Data” per fare update (in caso di conflitti l'utente dovrebbe poter chiedere di attivare i Dati se ha scelto “Only Data”), poiché su android >=5.0 sono state eliminate funzioni dal ConnectionManager (come la *setMobileDataEnabled)* e le uniche possibili funzioni da utilizzare (appartenenti al TelephonyManager) funzionano solo su smartphone con permessi di root.

e) Funzione di salvataggio file in locale, non implementata poiché, se i dati raccolti venissero postati in differita dal reale tempo di monitoring, il reporting (e le statistiche) non sarebbe né realistiche né veritiere (Thingspeak salva l'ora di post del valore).

Riferimenti

1 github.com/lecho/hellocharts-android
2 it.mathworks.com/help/thingspeak
3 developer.android.com/studio/index.html

Grazie dell'attenzione!