Emanuele Di Sante
emanuele.disante@studio.unibo.it
0000987616

Matilde Mariano
matilde.mariano@studio.unibo.it
0000970476

TransceiverGO

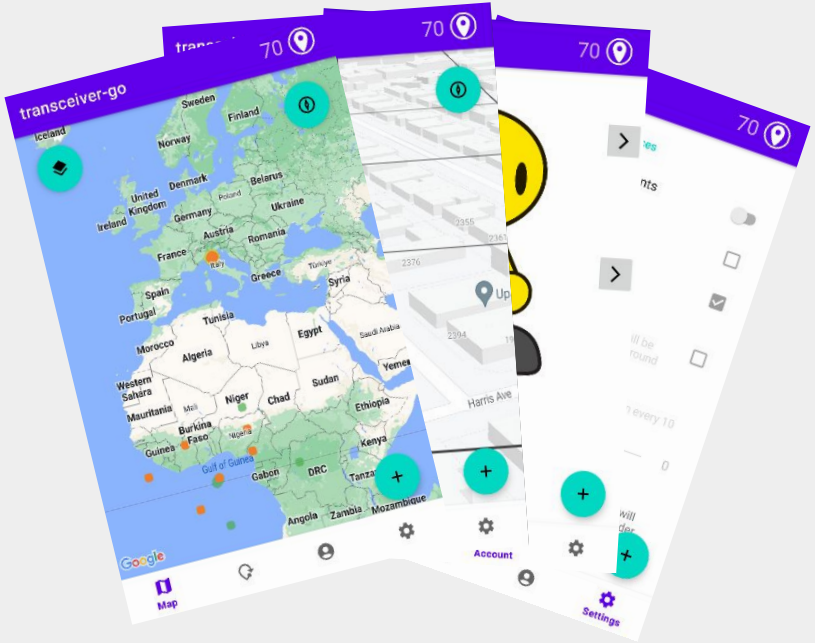
Progetto per il corso Laboratorio di Applicazioni Mobili
AA 2022/2023 Unibo

Overview

Interfaccia grafica

Raccolta e gestione dati

Overview dell’applicazione



- Raccoglie dati relativi alla connettività cellulare, intensità del segnale Wi-Fi e intensità del rumore contestualizzati in un’area geografica.
- Incentiva la raccolta dati tramite la componente di gioco.

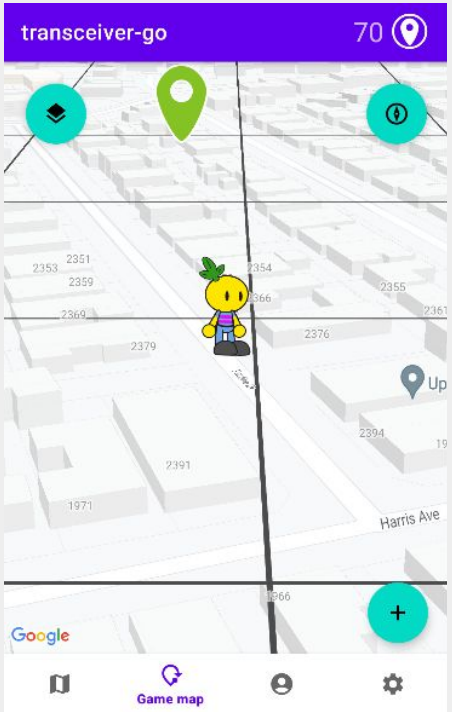
Overview

Interfaccia grafica

Raccolta e gestione dati

Due modalità di utilizzo

Visualizzazione dei dati



Gioco per la raccolta dati

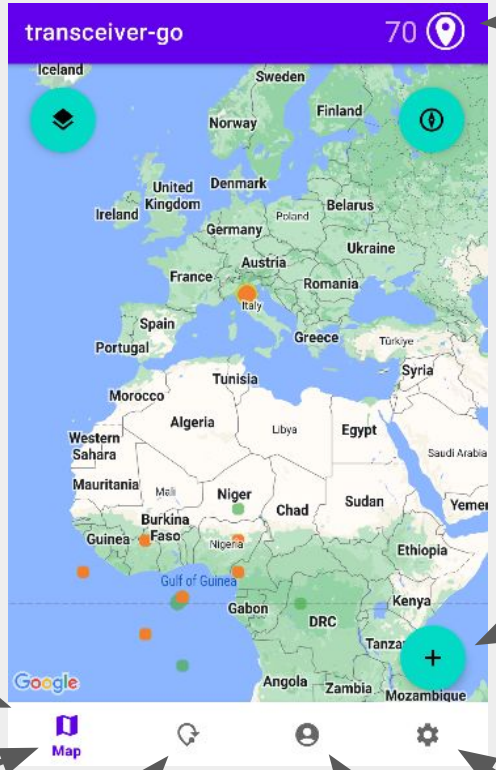
Guida all'uso

Overview

Interfaccia grafica

Raccolta e gestione dati

L'applicazione è disponibile in tre lingue:
- Inglese
- Italiano
- Spagnolo



Monete Transceiver

Barra di navigazione

Pulsante per aggiungere manualmente nuove registrazioni

Mappa

Gioco

Account

Impostazioni

Mappa

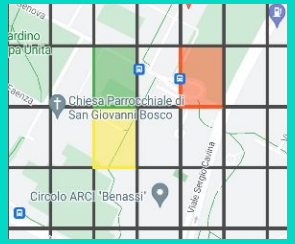
Overview

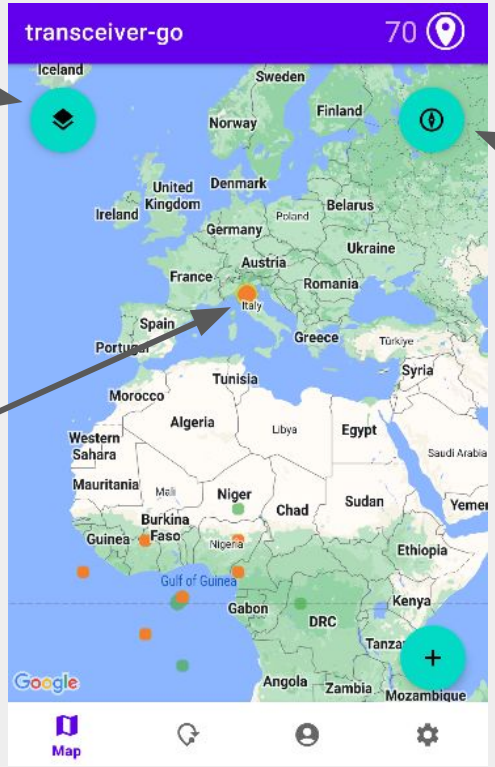
Interfaccia grafica

Raccolta e gestione dati

Pulsante per scegliere il tipo dei dati visualizzati ed i database da cui prenderli

Con alti livelli di zoom:





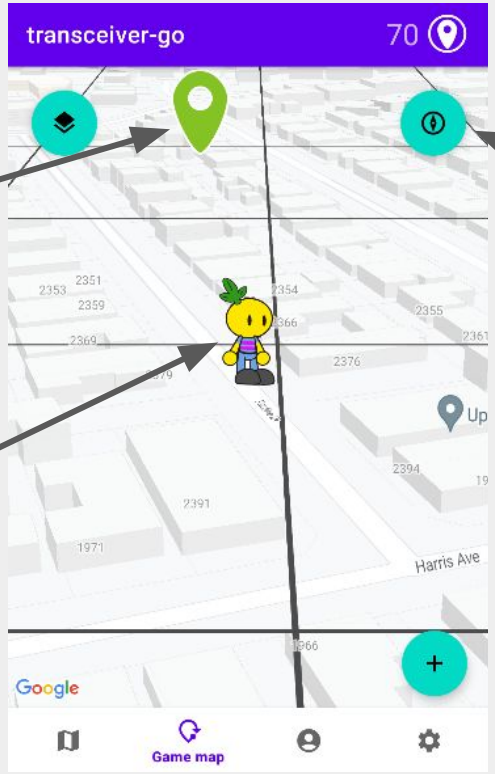
Resetta l'orientamento della mappa

- Overview
- Interfaccia grafica
- Raccolta e gestione dati

Gioco

Obiettivo che deve raggiungere il giocatore per ottenere la ricompensa

Avatar del giocatore



Centra la visuale sul giocatore, orientandola nella direzione in cui lui sta guardando

Overview

Interfaccia grafica

Raccolta e gestione dati

Gioco

La generazione dell'obiettivo è pensata in modo da spingere l'utente ad esplorare luoghi in cui non è mai andato o in cui non va da molto tempo.
Questo aiuta a popolare la mappa in maniera completa ed uniforme.

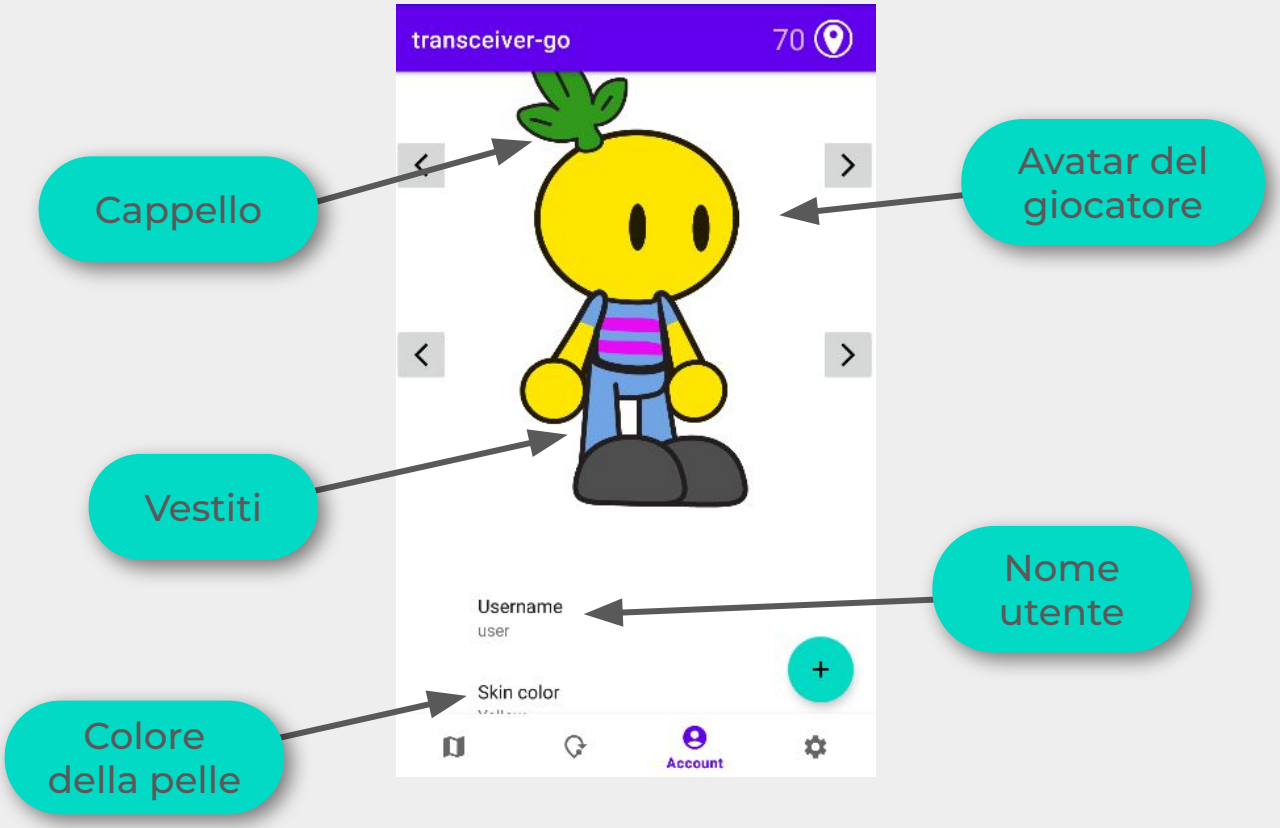
```
public Square generateNewTarget(List<Square> loadedSquares, String typeOfData){
    Collections.shuffle(loadedSquares); // randomizes the order of loadedSquares, bringing more entropy
                                        to the generation

    loadedSquares.sort(tipoDiDatoVisualizzato);

    int bound = Math.min(poolSize, loadedSquares.size());
    Random rand = new Random();
    int indexOfTarget = rand.nextInt(bound); // generates a random number between 0 and bound - 1

    typeOfDataUsed = typeOfData;
    currentTarget = loadedSquares.get(indexOfTarget);
    return currentTarget;
}
```

Overview

Interfaccia grafica

Raccolta e gestione dati

Account

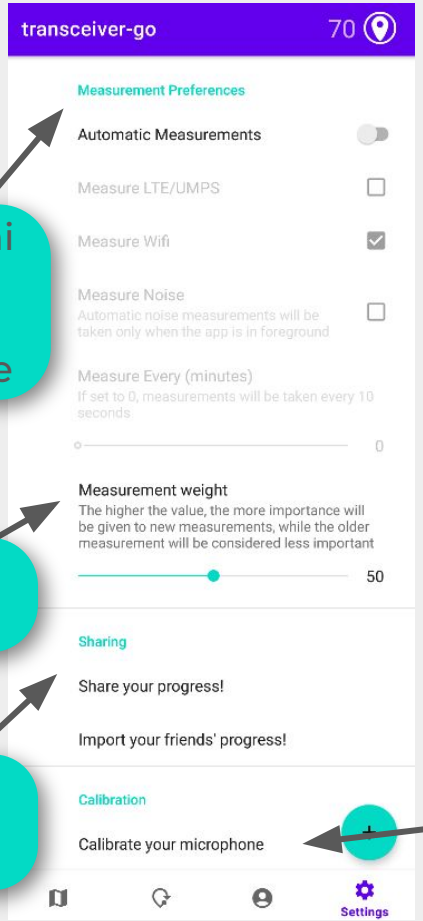


Overview

Interfaccia grafica

Raccolta e gestione dati

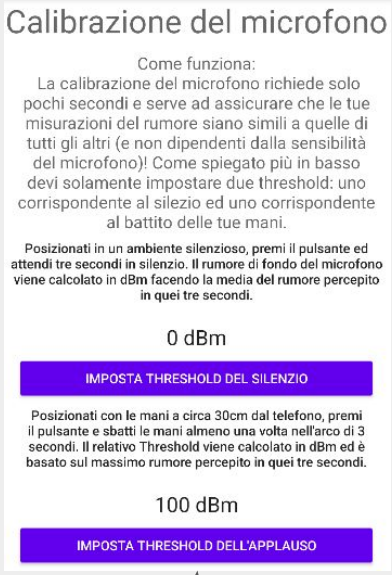
Impostazioni



Impostazioni delle misurazioni automatiche

Peso delle misurazioni

Condivisione dei database



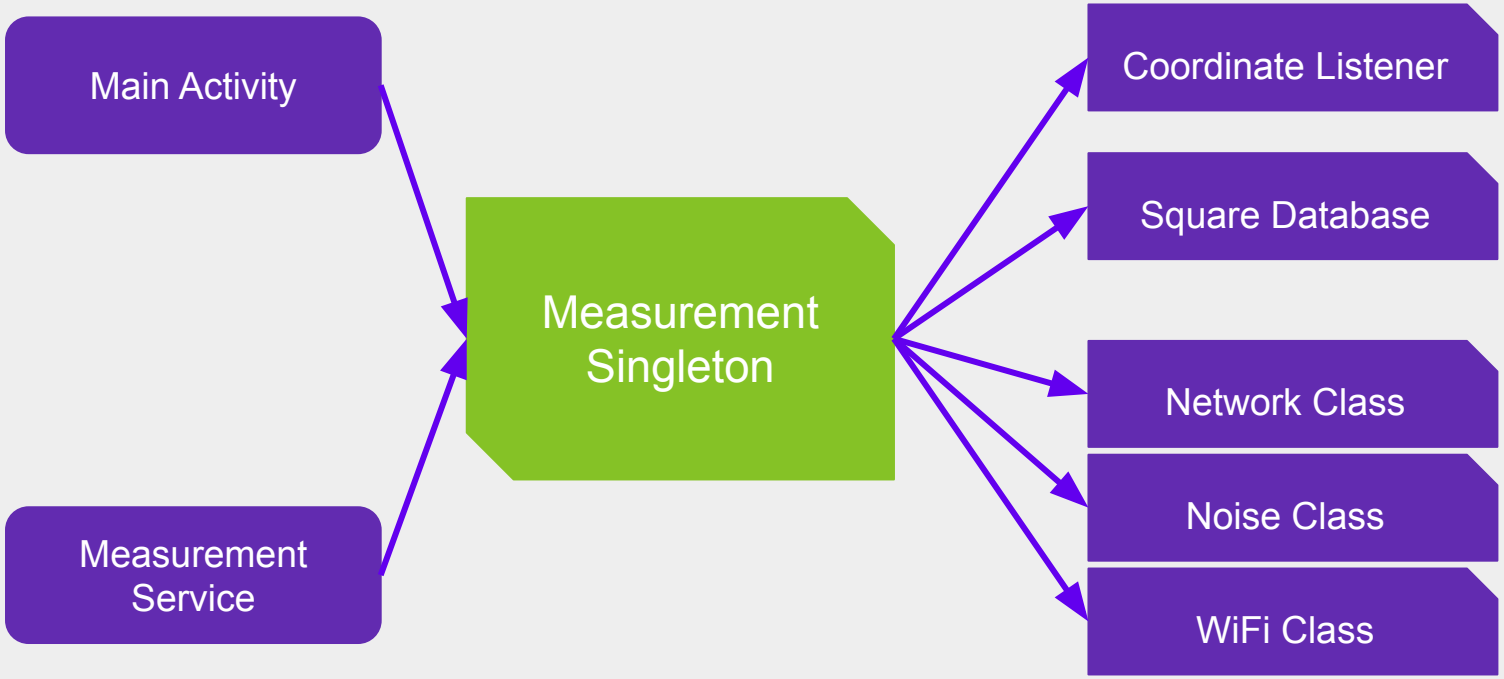
Questo pulsante porta ad un fragment per la calibrazione del microfono

Overview

Interfaccia grafica

Raccolta e gestione dati

Il punto centrale: Measurement Singleton



- La raccolta dei dati viene eseguita interamente dalla classe Measurement Singleton.
- Si interfaccia con tutti i componenti necessari per raccogliere e salvare i dati all'interno dell'applicazione.

Overview

Interfaccia grafica

Raccolta e gestione dati

Measurement Singleton

- Composto da due categorie di Metodi fondamentali:
  - take X measurement
  - update X measurement

```
public void takeXMeasurement(MainActivity activity) {
    // update current coordinates
    longitude = coordinateListener.getLongitude();
    latitude = coordinateListener.getLatitude();

    int newMeasurement = Sensor.getMeasurement();
    updateXMeasurement(newMeasurement);

  // refresh maps on screen to show the new measurement
    activity.refreshMaps();

  // notify the user
    Toast toast = Toast.makeText(R.string.taken_X_measurement);
    toast.show();
}
```

Overview

Interfaccia grafica

Raccolta e gestione dati

Measurement Singleton

- Composto da due categorie di Metodi fondamentali:
  - take X measurement
  - update X measurement*

```
private void updateXMeasurement(int newMeasurement) {
        //defining the square
        Square square = new Square(longitude, latitude);
        // returns the square we're in, if it exists
        Square squareInDb = squaredb.getSquare(square);
        // actual update
        square.updateX(newMeasurement);

        // update the database with updated square
        squaredb.upsertSquare(square);
        squaredb.close();
}
```

- Per le misurazioni Noise, un listener viene implementato

Overview

Interfaccia grafica

Raccolta e gestione dati

Measurement Service

```
private final Runnable measuringRun() {
        //check user preferences and take desired measurements
        if(network_measurement) {measurementSingleton.takeNetworkMeasurement();}
        if(wifi_measurement) {measurementSingleton.takeWifiMeasurement();}
        if(noise_measurement) {measurementSingleton.takeNoiseMeasurement();}

        // reschedule the task to run again in X minutes if needed
        if (automatic_measurements) {
            if (measure_interval == 0) handler.postDelayed(this, 1000 * 10); // 10 seconds
            else {
                handler.postDelayed(this, measure_interval * 60_000L);
             }
```

- Servizio che insieme alla Main Activity richiama il singoletto per le misurazioni
- Si occupa di gestire la notifica e di prendere misurazioni seguendo le preferenze utente

Overview

Interfaccia grafica

Raccolta e gestione dati

Classi Strength

- Si occupano di intermediare tra i sensori e restituire il valore elaborato raccolto in una pratica scala di misura.

- NoiseStrength
  - Implementa un Recording Listener per elaborare i dati a seguito della registrazione
- NetworkSignalStrength
  - Può ritornare sia il segnale UMTS che il segnale LTE
- WifiSignalStrength
  - Controlla il dispositivo abbia il Wifi abilitato e ritorna la qualità del segnale

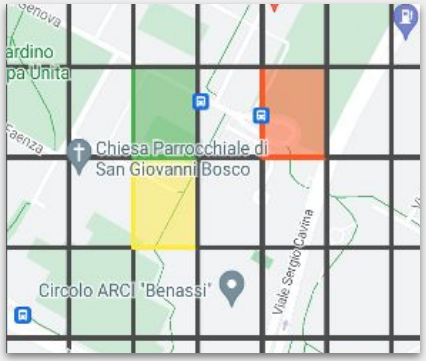
Overview

Interfaccia grafica

Raccolta e gestione dati

Square e il Database Room

- La classe Square rappresenta un quadrato sulla mappa.
- è progettata per:
  - gestire i dati relativi alle misurazioni di segnali nella sua area definita.
  - rappresentarsi a schermo sulle mappe con il relativo colore.



visualizzazione di alcune istanze della classe square

- Le istanze della classe square sono gestiti attraverso un database room definito come SquareDatabase.
- L'interazione con il database è gestita tramite il DAO (Data Access Object) definita sempre tramite Room.

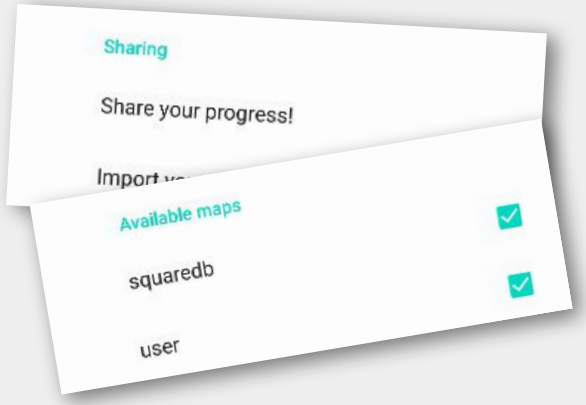
Overview

Interfaccia grafica

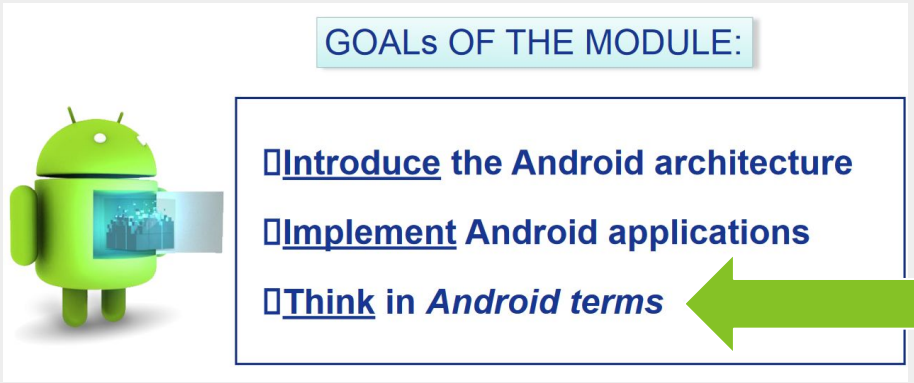
Raccolta e gestione dati

Importazione ed Esportazione

- La classe DatabaseImportExportUtil fornisce un insieme di utility per la gestione e la manipolazione dei database nell'applicazione a livello filesystem.
- La classe permette di:
  - Cambiare il nome ad un Database
  - Cancellare un Database
  - Esportare un Database (intent implicito)
  - Importare un Database



Fine presentazione




```
@Override
protected void onDestroy() {
    super.onDestroy();
    // chiediSeCiSonoDomande();
    this.chiudiPresentazione();
    }
```