Mimir technical documentation

Release-1.0

GNSS

GNSS data is collected in a GNSSHandler-class. There we use various APIs to get wanted data. Unfortunately, we still had to use deprecated methods to actually get what was wanted. Those are described later. To log all of the wanted GNSS data we need to use

1) LocationManager

- It is needed to register all of the listeners that react when a data collecting event appears

2) GNSSMeasurement and GNSSMeasurementEvent.Callback

- GNSSMeasurement is a class that has a lot of data from one event where GNSS data is collected. Unfortunately, some essential datapoints needed to be collected with deprecated functions. Those functions do work now but they could be removed in the future.

Here's the list of them and the functions the Android Studio suggests replacing them with:

- HasCarrierCycles
- HasCarrierPhase
- HasCarrierPhaseUncertainty
    - Replace with -> Use getAccumulatedDeltaRangeState

- HasAutomaticGainControlLevelDb
- GetAutomaticGainControlLevelDb
    - Replace with -> Use GnssMeasurementsEvent.getGnssAutomaticGainControls

- GNSSMeasurements are collected by creating a listener with abstract class GNSSMeasurementEvent.Callback. That has a function onGnssMeasurementReceived which is implemented to log data from one measurement. One measurement gives data from multiple satellites. We also get GNSSClock from the callback which also gives us data.

3) GNSSNavigationMessage and GNSSNavigationMessage.Callback

- Same way as GNSSMeasurement GNSSNAvigationMessage is a class that is used to get access to the received data and the callback is there to receive the data. They contain different data than GNSSMeasurement.

4) LocationListener

- Interface that gets implemented to receive information when phone's location changes. Listeners are registered to LocationManager in order to receive anything.

- These are created for both network and gps providers.

Logging GNSS and location requires the user to give permission for location data collecting. Permission is asked in the app.

Wifi

As of May 2023, logging wifi data is not implemented. Three different APIs were attempted for implementing fetching nearby wifi networks and getting their RSSI values. For various reasons, requirements were not able to be implemented with any of them. Below are listed the APIs and what the issue with them were.

1. Wifimanager
   - Deprecated
   - Unable to scan for nearby wifi networks without startScan, which is deprecated
     - StartScan deprecated in API level 28
2. WifiRttManager
   - Incapable of getting RSSI values
   - Only supports scanning for certain devices
     - Not all android devices are supported
   - RTT allows for measuring the time it takes for a Wi-Fi signal to travel between two devices.
3. ConnectivityManager
   - Can return RSSI value, but only for current network
   - Unable to scan for nearby wifi networks

Bluetooth Low Energy

Returns BLE device, RSSI value and scan record.

MotionSensorsHandler

Uses “SENSOR_SERVICE” from system services as sensor manager. This holds in the default sensors used for logging such as accelerometer (calibrated & uncalibrated), gyroscope (calibrated & uncalibrated), magnetometer, barometer and step counter. It also has many other sensors which we are not using but could be easily integrated.

ActivityHandler

Mobile phone version of Mimir has a singleton class called ActivityHandler, in which calling for sensors and logging happens. By accessing ActivityHandler it is possible to access the sensors.

SmartWatch

The smartwatch uses mostly similar classes and methods for the sensor data logging with similar functionalities.

File sending to phone happens from SendSurveysActivity using channelClient and sendFile function.

Before file is sent to smart phone, each of the chosen files are combined into one file which then gets sent.

WatchActivityHandler class is used to store filepaths of the files selected fir sending.

LoggingActivity is responsible of calling the sensorhandler classes and starting the logging process.

ReviewBtn activates the LoggedEvent class where user can choose to load or not to load the file.