MOVESENSE

MOVESENSE MOBILE PROGRAMMING

EDUCATION SESSION FOR CUSTOMERS

Petri Lipponen 2021-04-22

Content

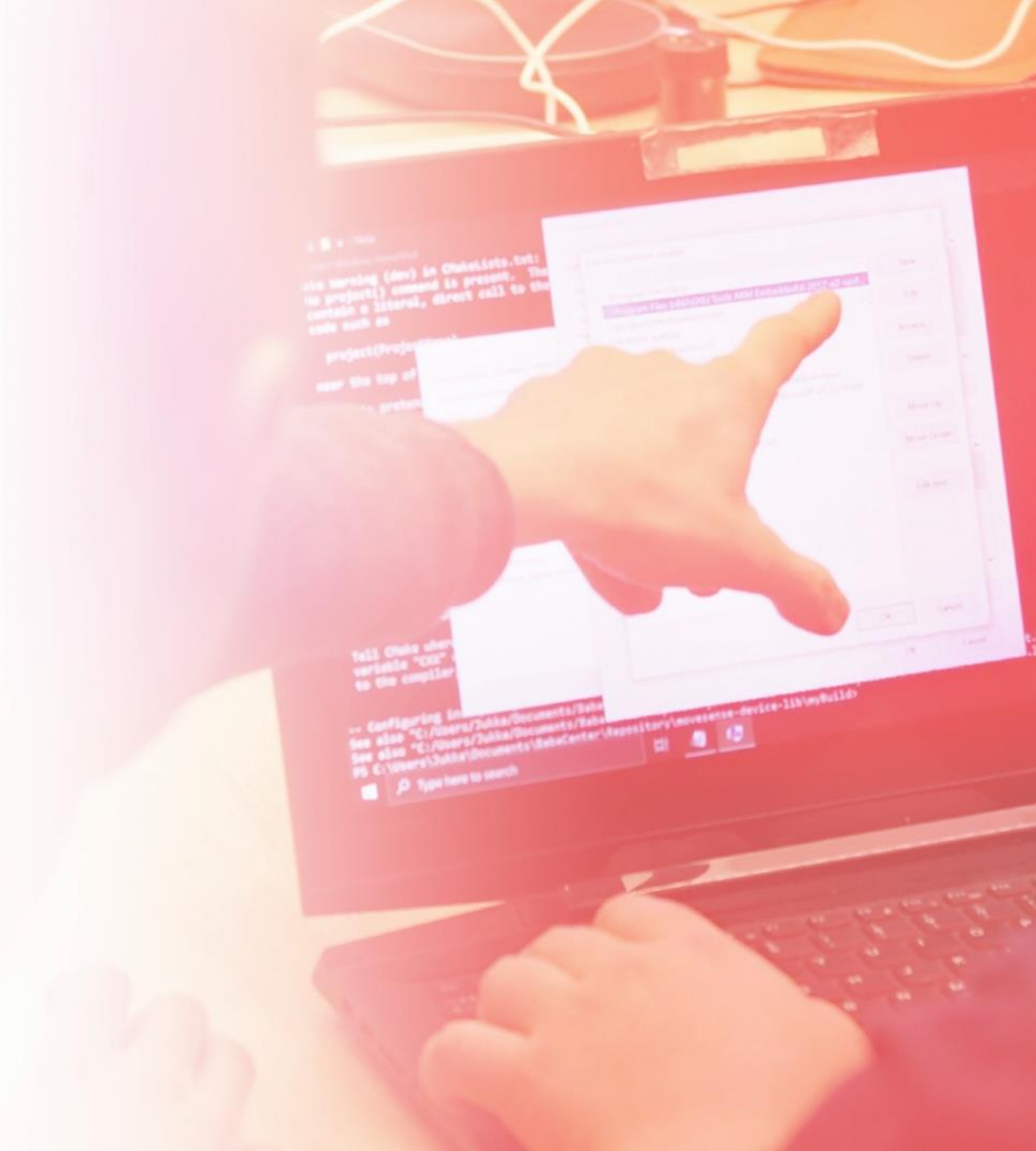
- What is Movesense?
- Movesense mobile architecture
- Movesense REST API
- Adding MDS into project (Android)
- Connecting to sensors (Android)
- Making MDS requests
- Updating sensor firmware
- Using data memory
- About iOS programming
- Q & A

“What is Movesense?”

In Short:

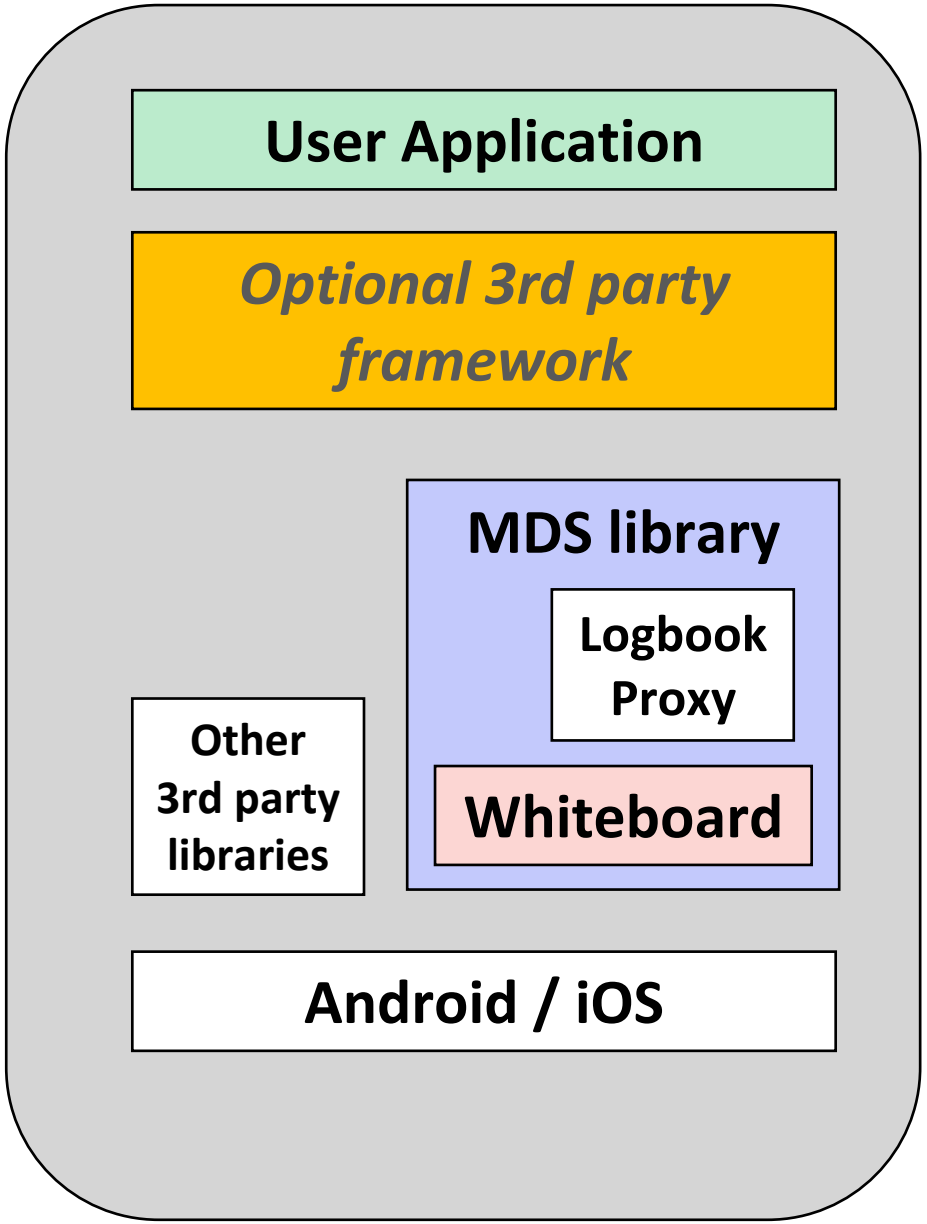
“Open Wearable Tech Platform”

- Programmable Sensor
- Mobile SDKs
- Embedded REST communication framework “Whiteboard”
- Physical accessories



Movesense Mobile Architecture

- Movesense Mobile library (aka. MDS)
  - Whiteboard REST communication to sensors
  - Lobgook proxy service
- Other 3rd party libraries
  - RxAndroidBle (Android BLE abstraction)
  - nRF DFU library (for firmware update)
- OS layer and libraries
- Possible other mobile frameworks from 3rd parties
  - Xamarin, Unity3D, ReactNative, Flutter, etc.



Movesense REST API

Main sections:

- */Meas* for sensor data (Acc, Gyro, Magn, Temp, HR, ECG)
- */Mem* for data memory access: DataLogger & Logbook
- */Comm* for communication protocols: BLE, 1Wire
- */Component* for low level features: LED, EEPROM, chip specific features
- */System for s*ystem features: Mode, Settings, Energy, Memory, States
- */UI* for user interface (LED blinks)
- */Misc* for the ones that do not fit in the above

and

- */Whiteboard* for Whiteboards own services

See: https://bitbucket.org/suunto/movesense-device-lib/src/master/MovesenseCoreLib/resources/movesense-api/

Movesense REST API: /Meas

Same pattern for all sensors

/Meas/*****/**Info**:

- Valid sample rates
- Sensitivity values (G-range etc.)

/Meas/****/**Config**:

- Sensitivity etc.

/Meas/**/<SampleRate>**:

- Data stream @ given sample rate (e.g. */Meas/Acc/13*)
- Note: when subscribing the /Subscribe is not given

Adding MDS Into your Project (Android)

- Add needed libraries into the gradle-file
  - MDS, RxAndroidBle, (GSON for parsing returned objects)
  - Copy from samples

- Create the MDS object (needs reference to context)
  - **The context needs to exist for the duration of operation!**

```
// RxAndroidBle
implementation "com.polidea.rxandroidble2:rxandroidble:1.10.2"
implementation 'io.reactivex.rxjava2:rxandroid:2.1.1'
implementation 'io.reactivex.rxjava2:rxjava:2.2.8'

// GSON
implementation 'com.google.code.gson:gson:2.8.0'

// Movesense .aar lib
implementation(name: 'mdslib', version: '1.44.0(1)-release', ext: 'aar')
```

```
private void initMds() {
    mMds = Mds.builder().build( context: this);
}
```

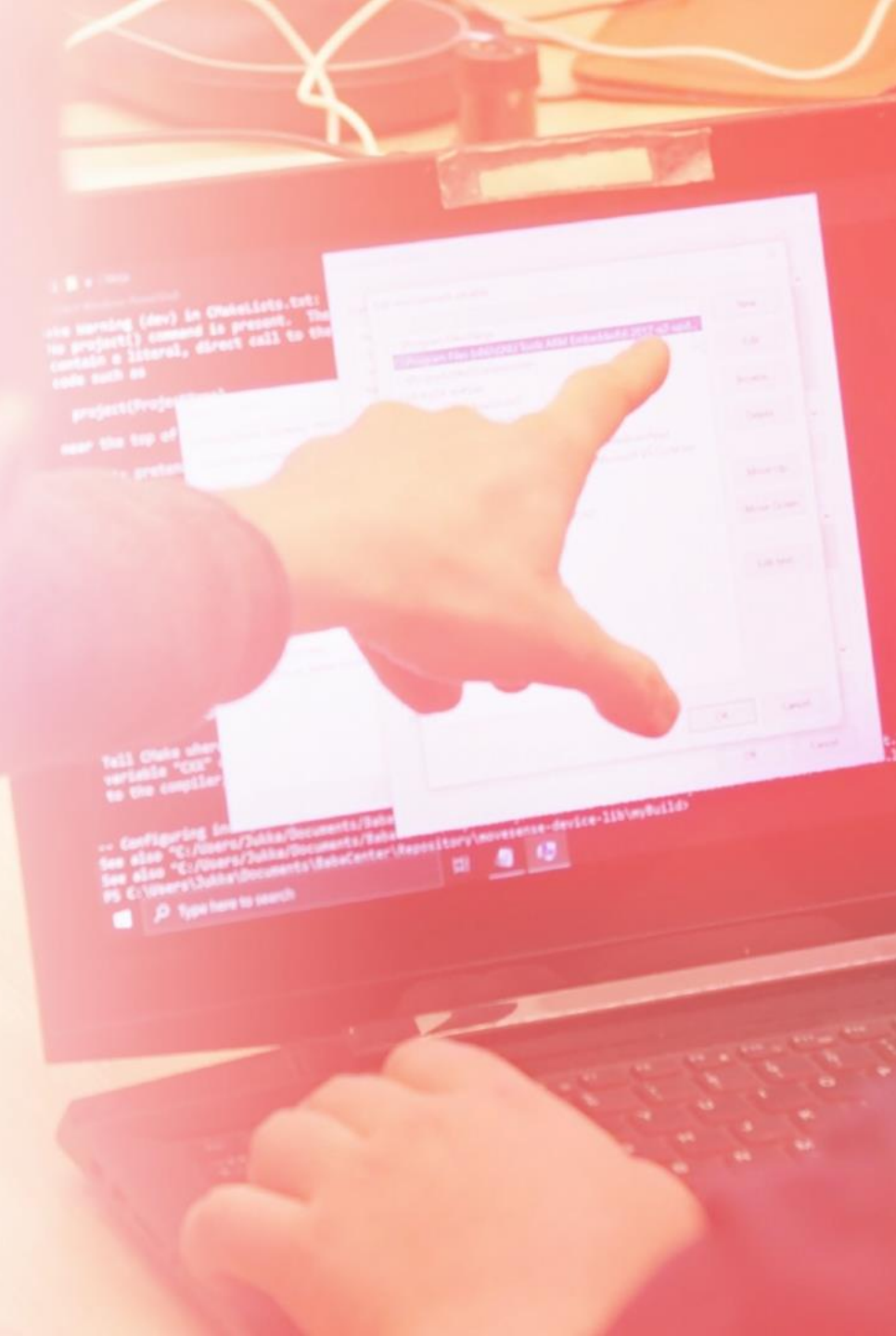
Connecting to Sensors (Android)

- Scan for devices unless deviceId (MAC address) is already known
  - Not part of MDS functionality

- Call *connect(deviceId, listener)* on MDS object
  - MdsConnectionListener handles the status callbacks

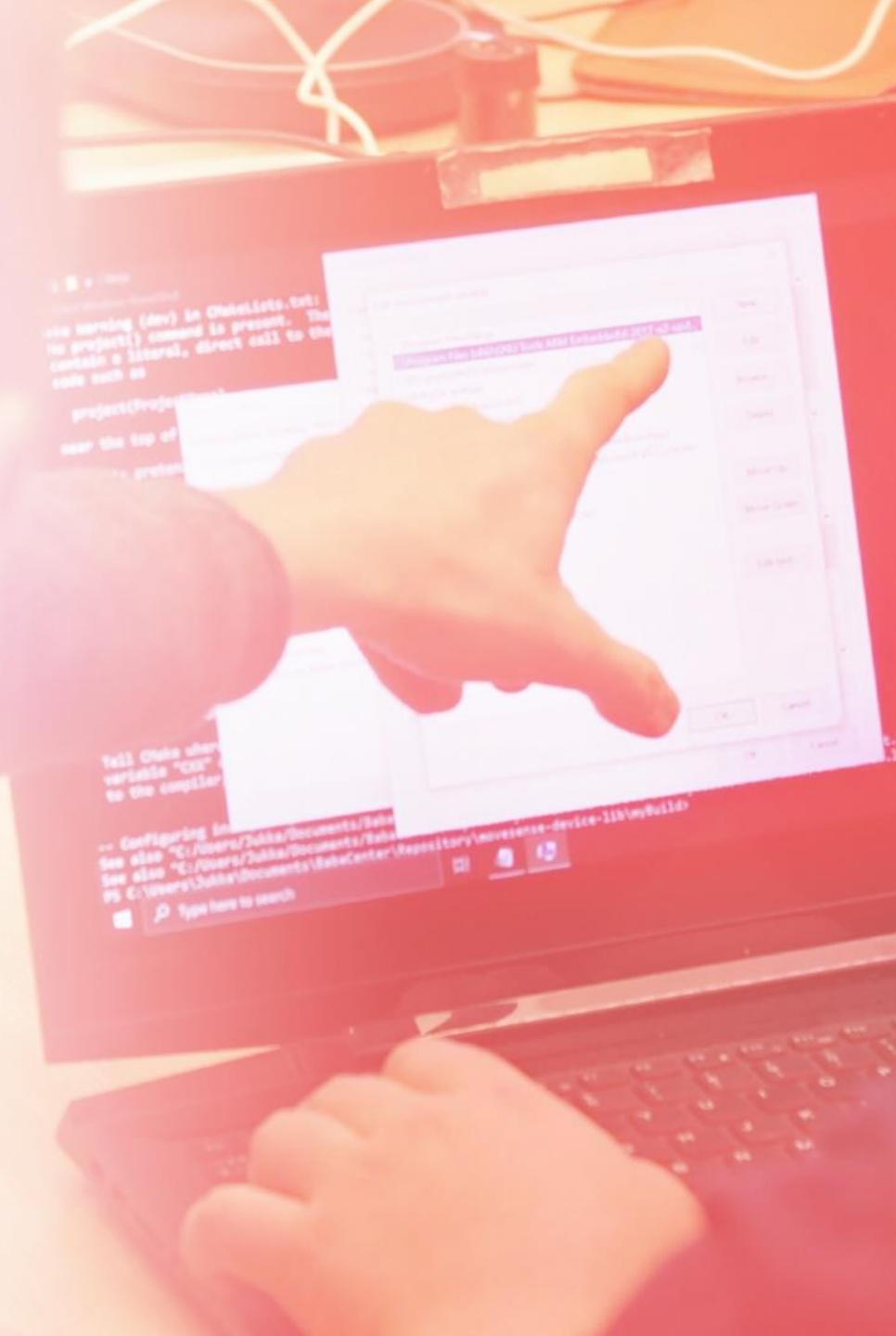
- Act on callbacks on the listener:
  - *onConnect*: BLE connection OK
  - *onConnectionComplete*: Whiteboard connection OK
  - *onError:* Error occured
  - *onDisconnect*: Disconnect happened.

    Note: If this was not ordered by call to *disconnet(deviceId)*, the MDS will try to reconnect automatically



Making MDS Requests: GET, POST, PUT, DELETE

- GET, POST, DELETE & PUT
  - REST verb in lowercase e.g: *mds.put(…)*
  - URL to the resource
  - Optional parameter object
  - Result of the request is returned in the *MdsResponseListener* callbacks
- URL format on sensor
  - `suunto://<device_serialnumber>/<resource_path>`
- Parameter object:
  - `{ ”paramName”:{<object as json>} }`
  - *paramName* & object format defined in API yaml definition
  - **EXAMPLES**
- MDS service URLs:
  - `suunto://MDS/...`



Making MDS Requests: SUBSCRIBE / UNSUBSCRIBE

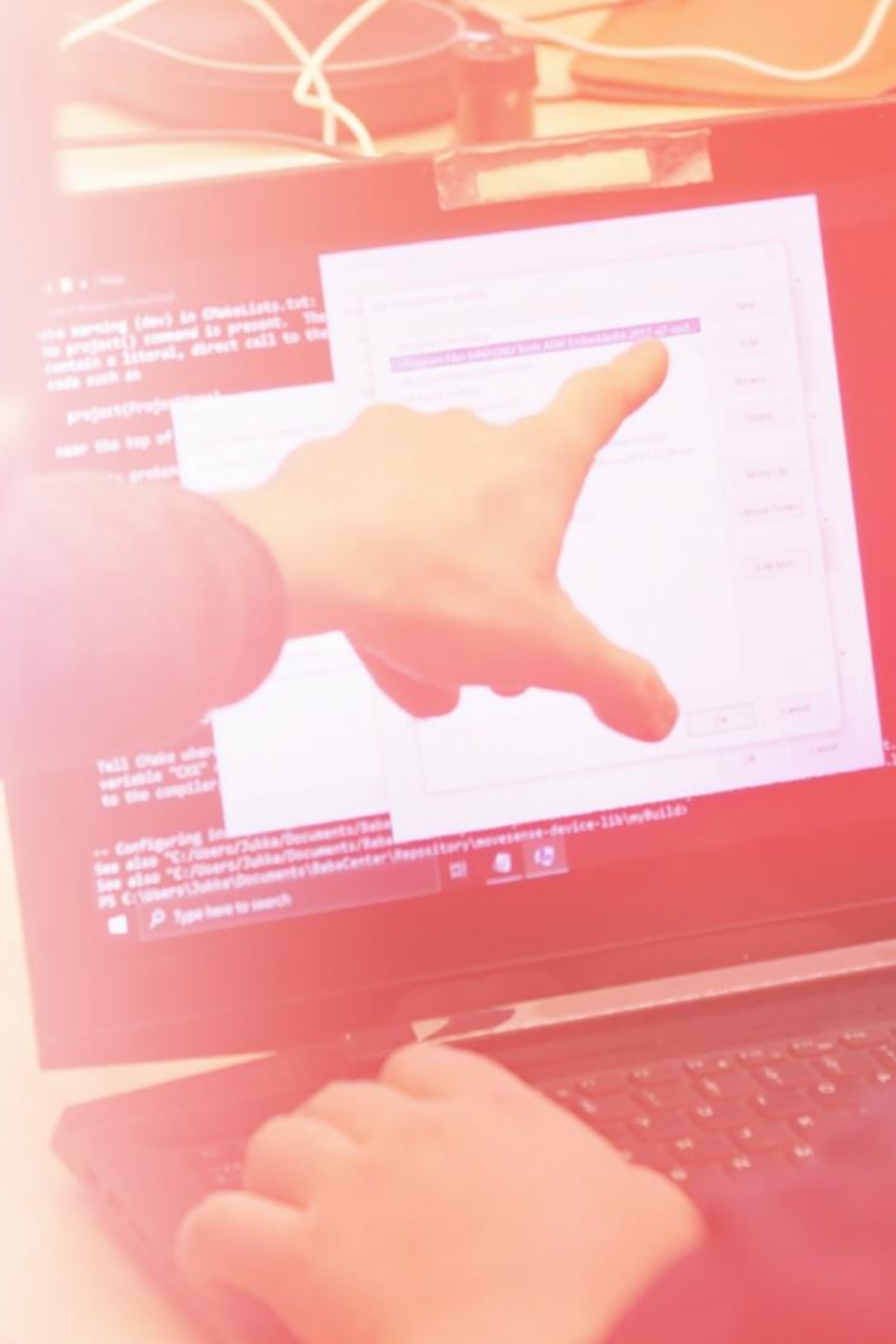
- Method *mds.subscribe(…)*
  - URI to event listener: `"suunto://MDS/EventListener"`
  - Contract object (json string):

    ```
    { "Uri": "<url to subscribe>" }
    ```

  - *MdsNotificationListener* object which receives the events in callbacks
  - Returns *MdsSubscription* object

- *MdsNotificationListener* Callbacks:
  - *onNotification(data):* event data as JSON string
  - *onError(…)* : Error happened

- To end subscription, call "*unsubscribe()*" on the *MdsSubscription* object
- **NOTE**: Subscriptions are invalidated if disconnect happens

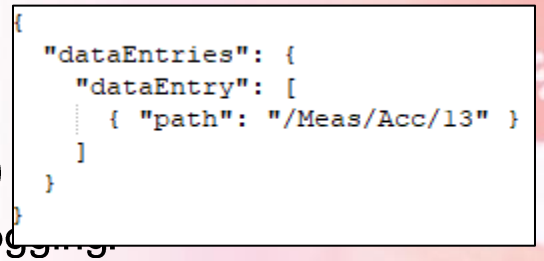


Updating Sensor Firmware

- Actual update is done by Nordic Semiconductors DFU library
- Sensor must be put to DFU mode:
  - PUT */System/Mode* with parameter: `{"NewState":12}`
  - DFU mode is indicated with LED ON and will either
    stay in it (<=v.1.9)
    or
    return to application mode after 2 minutes (>=2.0)
- Alternative way is ”DFU Recovery Mode”
  - http://movesense.com/docs/esw/dfu_update/

- DFU Libraries:
  - https://github.com/NordicSemiconductor/Android-DFU-Library
  - https://github.com/NordicSemiconductor/IOS-Pods-DFU-Library
  - https://github.com/Pilloxa/react-native-nordic-dfu
  - https://github.com/fengqiangboy/flutter-nordic-dfu

Using Sensor Data Memory: on-sensor services



- DataLogger on sensor
  - PUT */Mem/DataLogger/Config* to choose path(s)
  - PUT */Mem/DataLogger/State* to start and stop logging.
    - **3 = logging, 2 = not logging**

- Manage logs on sensor
  - GET */Mem/Logbook/Entries*: Enumerates Logbook content (4 at the time)
  - POST */Mem/Logbook/Entries*: Starts a new log
  - DELETE */Mem/Logbook/Entries:* Erases **whole** logbook content

- GET */Mem/Logbook/byId/<logID>/Data* and */Mem/Logbook/byId/<logID>/ Descriptors*:
  - *Returns the **binary content** of the log in SBEM format in small pieces* (HTTP_CODE_CONTINUE)

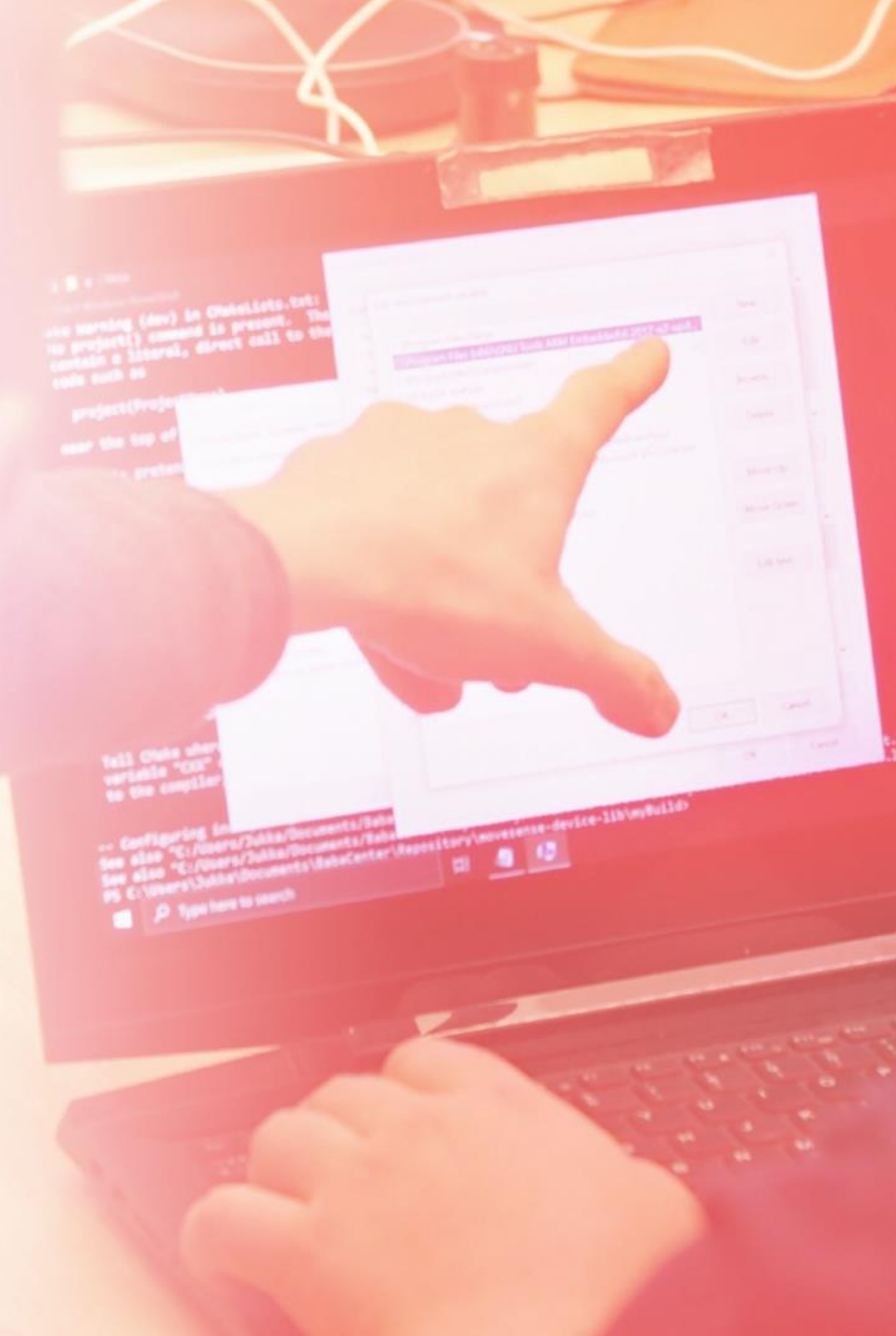
See:
http://www.movesense.com/docs/mobile/android/datalogger_logboo

Using Sensor Data Memory: MDS proxy service

Helper service that simplify Logbook use:

- GET *suunto://MDS/Logbook/<device_serial>/Entries*
  - Takes care of paging and retry in case of errors
  - Returns the whole datalogger entry collection as one JSON object

- GET *suunto://MDS/Logbook/<device_serial>/ById/<LogID>/Data*
  - Fetches **descriptors** and **data** from sensor
  - Handles retry logic in case of errors
  - Converts the SBEM binary and returns the full log as JSON string

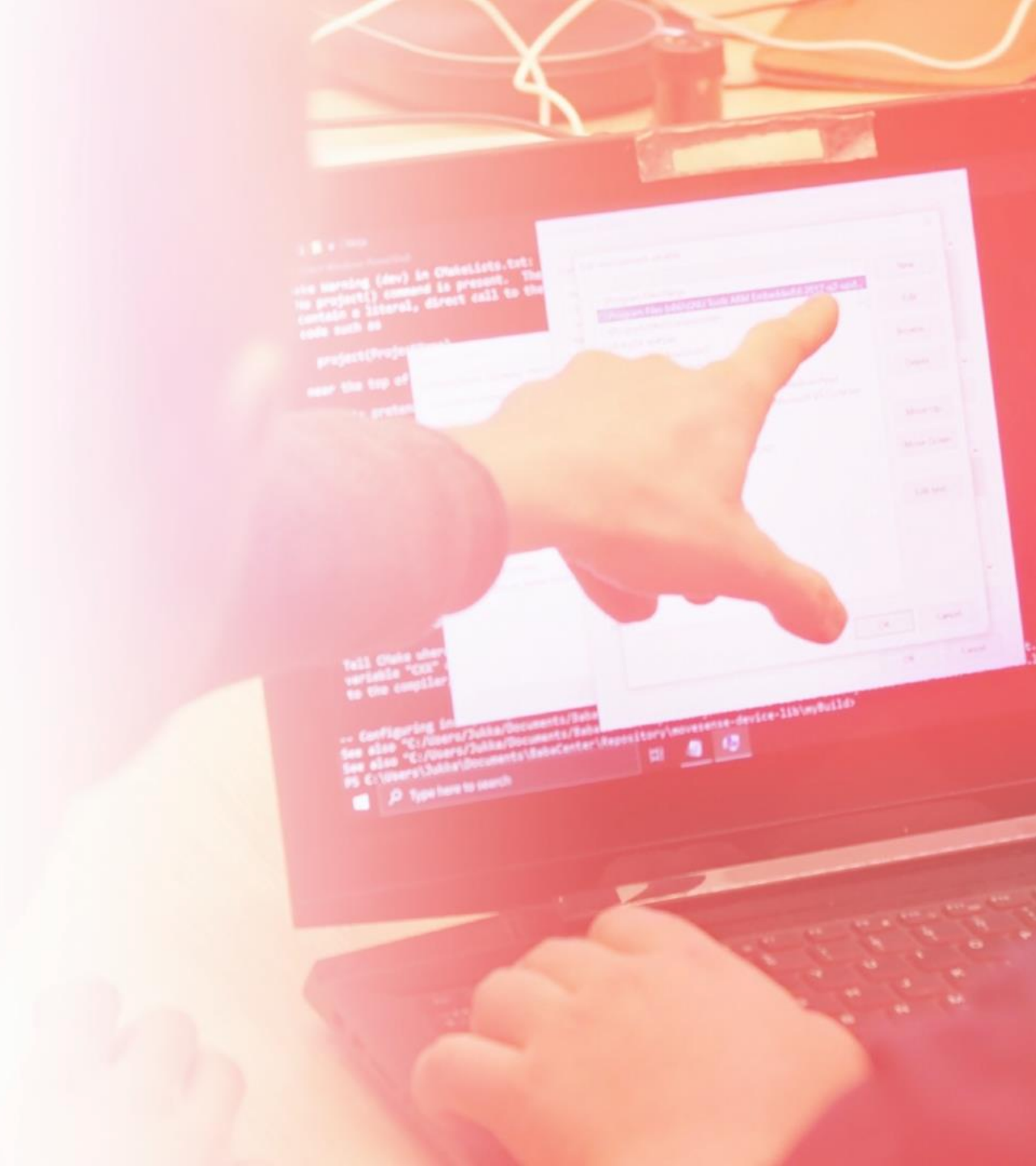


DataLoggerSample DEMO

https://bitbucket.org/suunto/movesense-mobile-lib/src/master/android/samples/DataLoggerSample/

About iOS programming

- Start with Movesense Showcase app: https://bitbucket.org/suunto/movesense-mobile-lib/src/master/IOS/MovesenseShowcase/
- REST API calls very similar to Android
- MDS for watchOS not (yet) available



MOVESENSE

Questions?