USER GUIDE

How to install and use the My Brain Technologies’ SDK in your projects

# General Overview

This document aims at helping users to install and use the My Brain Technologies’ SDK in their own Android project. It is arranged in a way to guide you through the various steps in your development process. We recommend following each of the steps outlined below and reading the documentation in the order displayed.

The SDK is a closed source library developed by My Brain Technologies. Its main purpose is to allow the development of external applications based on My Brain Technologies’ headset. It also provides code samples, and technical and use case documentation for developers.

My Brain Technologies’ headset includes Melomind and VPro headsets:

* Melomind is an audio headset composed of 2 electrodes that record electrical activity of the brain. These 2 channels of EEG acquisition are located on P3 and P4 parietal position.
* VPro is a headset composed of 8 electrodes that record electrical activity of the brain. These channels of EEG acquisition can be located at customized position of the brain

This library is currently compiled as a .aar file. Its content is obfuscated. Only the public content is accessible to external applications.

# Versions and features

## Requirements & configurations

Using the My Brain Technologies’ SDK requires to install an IDE for developing Android applications. *Warning: this document explains how to install the SDK on Android Studio IDE only.*

The My Brain Technologies’ SDK is compiled with Android API version 27 and is compatible with Android API version 22 and higher. Older versions will not work.

Please adjust your application’s minimum API version accordingly.

## Features

The current version of the SDK is 2.0.0. The SDK supports the following features: (in grey are features under development and not available yet)

Bluetooth Features:

* Bluetooth Low Energy connection with Melomind headset
* Bluetooth Low Energy disconnection with Melomind headset.
* Bluetooth Serial Port Profile connection with the VPro headset.
* Bluetooth Serial Port Profile disconnection with the VPro headset.
* Starting streaming to receive EEG data acquired from the headset.
* Stopping streaming to stop receiving EEG data

EEG Data Processing Features:

* Processing of EEG raw data acquired by the headset including
  + Conversion of EEG raw data acquired into user-readable EEG data values.
  + Notification from SDK to User Application that returns the EEG converted values.
  + Computation of the resulting status corresponding to an EEG data.
  + Computation of the signal quality for each channel of EEG acquisition of the headset.
  + Computation of the resulting relaxation level corresponding to an EEG data.
  + Computation of the results/score based on the relaxation indexes of a given exercice.

EEG Data Recording Features:

* Recording of EEG data acquired by the headset including
  + Starting a new recording that contains user-readable EEG data.
  + Stopping a current EEG data recording
  + Storage of the data recordings into JSON files.
  + Sending of the JSON files that contains EEG data to the server.

Further updates will be released with more content.

# How to install the SDK on your project

1. Create an account at TODO: insert address here.

Keep your login and password safe. You’ll need them to access to the library

1. Inside your gradle.properties file, add the following lines:

nexusUrl=https://myurl.com/repository/maven-releases/  
nexusUsername=username  
nexusPassword=password

1. Replace the username and password with your own credentials
2. Inside your app build.gradle file, add the following block of code

repositories{  
 maven{  
 url nexusUrl  
 credentials {  
 username nexusUsername  
 password nexusPassword  
 }  
 }  
}

1. Add the following dependency to your dependencies list:

implementation 'com.mybraintech:mbtsdk:2.0.0

1. Build with ./gradlew build command or “Sync now” option on Android Studio.
2. That’s it, the SDK should be available.

# How to use the SDK

## Requirements

The SDK requires min Android version to **22**.

The SDK is always targeting the latest version of Android SDK. Latest version is 27.

The SDK relies on Bluetooth Low Energy scanner. In order to access it, you need to request the following runtime permissions for devices running Android 6.0 and superior:

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE"/>  
<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION"/>  
<uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION"/>

To do so, you must request permission when you want the user to connect to a remote Bluetooth device.

## Communication between SDK & application

Once the SDK is installed on the application project, the users can use the several methods for developing their own application. The SDK communicates with the application through a client: the MbtClient.

The first step is to create a MbtClient instance and initialize this client inside the OnCreate method of the main activity of the application.

private MbtClient client;

@Override  
protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);

MbtClientEvents mbtClientEvents = new MbtClientEvents();client = MbtClient.*init*(getApplicationContext(),mbtClientEvents);  
}

When a MbtClient instance is initialized, it constructs a MbtManager object, that provides the user access to the Bluetooth and EEG processing methods.

The application is now set up to interact with the SDK features.

## Bluetooth Features

The Bluetooth communication between the headset and the application is managed by the MbtBluetoothManager. This object allows Bluetooth devices detection, connection, disconnection, data streaming, reading informations about the device (headset) and configuring the headset.

client.scanDevicesForType(deviceType, duration, scanCallback)

client.connectBluetooth()

client.disconnectBluetooth()

client.configureHeadset()

client.readBattery()

client.stopReadBattery()

1. EEG Features

For starting the EEG acquisition, the user needs to call the following method:

client.startStream(streamConfig)

This method starts to transmit the EEG data acquired by the headset to the application.

Parameters

StreamConfig streamConfig is the streaming configuration object, that includes the notification period //todo compléter

For using the EEG signal quality computation feature, the user needs to call the following method:

client.stopStream()

For using the EEG signal quality computation feature, the user needs to call the following method:

client.computeEEGSignalQuality(sampRate, packetLength, channels)

Parameters

int sampRate is the sampling rate

int packetLength is the length of a EEG packet(=time x sampRate)

Float[] channels is the array that contains the EEG channels of acquisition

For using the EEG relaxation index computation feature, the user needs to call the following method:

client.computeRelaxIndex(sampRate, calibParams, packets)

Parameters

int sampRate is the sampling rate

MbtCalibrationParameters calibParams is the object that contains the calibration parameters previously performed

MbtEEGPacket packets is the object that contains EEG data, theirs status and qualities.

For using the results computation feature based on the relaxation indexes of a given exercice, the user needs to call the following method:

client.computeResults(threshold, relaxIndexValues);

Parameters

float threshold is the level that corresponds to a relaxation state. Under this threshold, a computed relaxation index indicates that the subjct is in a non relaxed state.

Float[] relaxIndexValues is the array that contains all the computed relaxation indexes of a given relaxation exercice.

1. Recording Features

The EEG data recording is managed by The MbtRecordingSessionManager. This object allows recording the EEG data acquired : a recording is used to save these data into internal JSON files, and sending these JSON files to the server for an external save in the database.

For starting recording the EEG data, the user needs to call the following method:

client.startRecord();

Parameters

No parameters

For stopping recording the EEG data, the user needs to call the following method:

client.stopRecord();

Parameters

No parameters

For saving EEG data recordings in a JSON file, the user needs to call the following method:

client.saveRecordIntoJSON();

Parameters

No parameters

For saving JSON files, the user needs to call the following method:

client.saveRecordIntoJSON();

Parameters

No parameters

1. Server Features

