AOJ Bluetooth Device SDK Instruction

Android

Version 1.0.0 beta18

Catalog

1.Version Information

1.1.Update record

1.0.0． Version1.0.0

Add basic function implementation and interface support for product AOJ-20A、AOJ-30B、AOJ-70B

2.Operating environment

2.1. System requirements

The Android system is required to be above 4.3, and the mobile phone hardware must support Bluetooth 4.0

2.2. Permissions Settings

Add the following permissions to the AndroidManifest.xml of the project

```
// Basic Bluetooth application permission
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />

//Android 6.0 + need to add location permission
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>

// When enabling the log logging feature of SDK, add the following permission
<uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />

<uses-permission android:name="android.permission.BLUETOOTH_SCAN"/>
<uses-permission android:name="android.permission.BLUETOOTH_CONNECT"/>
```

3.Interfaces

3.1. Definition

3.1.1. getInstance

Static method, get the AHDevicePlugin object instance.

Parameter: None

Return: AHDevicePlugin.

3.1.2. initPlugin

Object instance initialization.

Parameter:

①Context appContext

Return: boolean

3.1.3. isSupportBLE

Check if the mobile phone Bluetooth supports low power consumption.

Parameter: None

Return: boolean.

3.1.4. isBluetoothAvailable

Check if the mobile phone Bluetooth is turned on.

Parameter: None

Return: boolean.

3.1.5. getManagerStatus

Gets the SDK's current working status.

Parameter:None

Return: BTManagerStatus.

3.1.6. getVersion

Get the current version information for the SDK

Parameter:None

Return: String

3.1.7. unregisterReceiver

Cancel the registered Bluetooth broadcast receiver

Parameter:None

Return: String

3.1.8. registerReceiver

Register for the Bluetooth broadcast receiver

Parameter:None

Return: String

3.1.9. searchDevice

Search for nearby Bluetooth devices

Parameter:

①ScanFilter filter,

②OnSearchingListener listener, search results listener

Return: boolean.

3.1.10. stopSearch

Stop search for nearby Bluetooth devices

Parameter: None

Return: boolean.

3.1.11. setDevices

Reset or clear the SDK measurement device list.

Parameter:List<BTDeviceInfo> list

Return: boolean.

3.1.12. addDevice

Add a single measuring device.

Parameter: BTDeviceInfo device

Return: boolean.

3.1.13. removeDevice

Disconnect the target device's bluetooth connection and remove the device from the SDK device list cache.

Parameter:String broadcastId.

Return: boolean.

3.1.14. startAutoConnect

Start the automatic connection synchronization service of the measurement device, and the interface supports the automatic reconnection mechanism.

The App does not need to handle the disconnected reconnecting, and it is not recommended to call the interface frequently.

Parameter:

① OnSyncingListener listener.

Return: boolean.

3.1.15. stopAutoConnect

Stop the automatic synchronization of device data and disconnect all established Bluetooth connections.

Parameter: None

Return: boolean.

3.1.16. resetSyncingListener

Reset the device data synchronization callback object

Parameter:

①OnSyncingListener listener.

Return: boolean.

3.1.17. checkConnectState

Get the connection status of the specified device

Parameter:

①String broadcastId.

Return: BTConnectState.

3.1.18. pushSetting

Device setting information update method，such as temperature measurement mode switching, user switching, etc

Parameter：

①String broadcastId

②**BTDeviceSyncSetting syncMsg, abstract class, please refer to the definition of the subclass**

③OnSettingListener listener

Return：void

3.1.19. saveDebugMessage

Enable the SDK log file logging feature

Parameter：

①boolean status , logging status, true on, false off

②String logPath, log file storage path, using the absolute path

③String appVersion

Return：void

3.1.20. appendlog

In the log log file, write the custom log information

Parameter：

①String msg,

Return：void

3.1.21. setManagerConfig

Connection parameters, scan interval setting method

Parameter：

①IBManagerConfig config

Return：void

4.Callback

4.1.OnSearchingListener .

4.1.1. onSearchResults

void onSearchResults(BTDeviceInfo device)

Device Bluetooth scan result callback

Parameter:

① BTDeviceInfo device

Return: none

4.1.2. onSystemBondDevice

void onSystemBondDevice(BluetoothDevice device)

System paired devices result callback

Parameter:

①BluetoothDevice device

Return: void

4.2. OnSettingListener

4.2.1. onSuccess

void onSuccess(String macAddress)

Set success callback

Parameter:

① String macAddress

4.2.2. onFailure

void onFailure(int errorCode)

Setting failed callback.

Parameter:

① int errorCode

4.2.3. onDataUpdate

void onDataUpate (Object obj)

Device Data callack.

Parameter:

① Object obj

4.6.OnSyncingListener.

4.6.1. onStateChanged

void onStateChanged (String broadcastId, BTConnectState state)

Device connection state changes callback

Parameter:

①BTConnectState state.

② String broadcastId.

Return: void

4.6.2. *onDeviceDataUpdate*

void onDeviceDataUpdate (String broadcastId,IDeviceData obj)

Device measurement data and status callback.

Parameter:

① String broadcastId.

② IDeviceData obj **abstract class, please refer to the definition of the subclass**

Return: void

5. Instructions

5.1. Initialization

1、Imports the SDK's Jar file into the lib folder of the Android Studio project

2、Config build.gradle, add dependencies

```
dependencies {
      implementation fileTree(dir: 'libs', include: ['*.jar'])
}
```

3、Adds the related application permissions requests in the AndroidManifest.xml

4、init AHDevicePlugin

```
public class MyApplication extends Application {
    @Override
    public void onCreate(){
        super.onCreate();
        //init AHDevicePlugin
        AHDevicePlugin.getInstance().initPlugin(getApplicationContext());
    }
}
```

5.2. Device Scan

Step 1、register scan results listener

```
OnSearchingListener listener=new OnSearchingListener() {
      @Override
      public void onSearchResults(BTDeviceInfo device) {
      //TODO
      //handleScanReults(device);
   }
};
```

Step 2、check manager status

```
BTManagerStatus sdkStatus=AHDevicePlugin.getInstance().getManagerStatus();
if(sdkStatus == BTManagerStatus.Free){
 //scan all device
 AHDevicePlugin.getInstance().searchDevice(null, listener);

 //scan Blood Pressure Meter & Oximeter
 List<BTDeviceType> types=new ArrayList<BTDeviceType>();
 types.add(BTDeviceType.BloodPressureMeter);
 types.add(BTDeviceType.Oximeter);

 ScanFilter filter = new ScanFilter(types);
 AHDevicePlugin.getInstance().searchDevice(types, listener);
 }
 else{
      //TODO
 }
```

5.3. Device Connect & Data Sync

Step 1、add device

App needs to fill in the device MAC, BroadcastID, DeviceType when adding devices.
Reference examples are shown below

```
String deviceMac="FA:B2:CA:4A:9A:94";
BTDeviceInfo device=new BTDeviceInfo();
device.setBroadcastID(deviceMac.replace(":",""));
device.setMacAddress(deviceMac);
device.setDeviceType(BTDeviceType. Oximeter.getValue());

AHDevicePlugin.getInstance().addDevice(device);
```

Step 2、register listener

```
final OnSyncingListener listener=new OnSyncingListener() {
    @Override
    public void onStateChanged(String broadcastId, LSConnectState state) {
        super.onStateChanged(broadcastId, state);
    }

    @Override
    public void onDeviceDataUpdate(String mac,IDeviceData obj){
        super. onDeviceDataUpdate (mac, type, data);
    }
};
```

Step 3、check manager status

```
BTManagerStatus sdkStatus=AHDevicePlugin.getInstance().getManagerStatus();
if(managerStatus == BTManagerStatus.Scanning){
    //取消扫描(stop search)
    AHDevicePlugin.getInstance().stopSearch();
}

if(sdkStatus == BTManagerStatus.Syncing){
    // In the Syncing state, the App does not need to call startAutoConnect repeatedly, and does not need to process the reconnection after disconnection. There is an automatic reconnection mechanism inside the SDK, and the App does not need to process this state.
    return ;
}
```

Step 4、start device data sync

```
if(sdkStatus == BTManagerStatus.Free){
    AHDevicePlugin.getInstance().startAutoConnect(listener);
}
```

Step 5、stop device data sync

```
    //If the application exits or needs to be disconnected, please call this method
    AHDevicePlugin.getInstance().stopAutoConnect();
```

5.5. Device Settings

All device settings, or data query, data deletion use the same method with different parameters

```
// Register or listen to set callback results
OnSettingListener listener = new OnSettingListener() {
    @Override
    public void onSuccess(String macAddress) {
        super.onSuccess(macAddress);
    }

    @Override
    public void onFailure(int errorCode) {
        super.onFailure(errorCode);
    }
};

/**
 * setting, abstract class
 * different settings need to initialize specific instance objects
 */
AHDevicePlugin.getInstance().pushSetting(mac, setting, listener);
```

5.5.1 Temperature measurement mode settings

Modify the thermometer temperature measurement mode or unit, examples are as follows

```
AHTempSetting setting = new AHTempSetting(AHTempCmd.ConfigMode);

/**
 * Adult(1),Children(2),Ear(3),Material(4);
 */
setting.setMode(AHTempMode.valueOf(value));

/**
 * 0x00 = Celsius
 * 0x01 = Fahrenheit
 */
setting.setUnit(item.getIndex());
```

5.5.2 Thermometer historical data synchronization

New Firmware Device

```
AHTempSetting setting = new AHTempSetting(AHTempCmd.NewSyncData);
```

Old Firmware Device

```
AHTempSetting setting = new AHTempSetting(AHTempCmd.SyncData);
```

5.5.3 Thermometer time synchronization

New Firmware Device

```
AHTempSetting setting = new AHTempSetting(AHTempCmd.NewSyncTime);
```

Old Firmware Device

```
AHTempSetting setting = new AHTempSetting(AHTempCmd.SyncTime);
```

5.5.4 Thermometer start measuring

```
AHTempSetting setting = new AHTempSetting(AHTempCmd.NewStartMeasuring);
```

5.5.5 Thermometer data remove

```
AHTempSetting setting = new AHTempSetting(AHTempCmd.ClearData);
```

5.5.6 Thermometer status sync settings

```
AHTempSetting setting = new AHTempSetting(AHTempCmd.QueryStatus);
```

5.5.7 Blood pressure meter offline data sync settings

Sync offline data based on device user

```
/**
 * 1 = Device User 1
 * 2 = Device User 2
 */
AHBpmSyncSetting setting = new AHBpmSyncSetting(1);
```

Sync all offline data

```
/**
 * true = Get all user data
 */
AHBpmSyncSetting setting = new AHBpmSyncSetting(true);
```

5.5.8 Blood pressure meter status sync settings

```
AHBpmConfigSetting  setting= new AHBpmConfigSetting(AHBpmConfig.StatusSync)
```

5.5.9 Blood pressure meter get device’s sn

```
AHBpmConfigSetting  setting= new AHBpmConfigSetting(AHBpmConfig.GetSn)
```

5.5.10 Blood pressure meter power off

```
AHBpmConfigSetting  setting= new AHBpmConfigSetting(AHBpmConfig.PowerOff)
```

5.5.11 Blood pressure meter data remove

Remove data based on device user

```
/**
 * 1 = Device User 1
 * 2 = Device User 2
 */
AHBpmRemoveSetting setting = new AHBpmRemoveSetting(1);
```

Remove all user data

```
/**
 * true = Remove all user data
 */
AHBpmRemoveSetting setting = new AHBpmRemoveSetting(true);
```

5.5.12 Blood pressure meter user switching

```
/**
 * 1 = Device User 1
 * 2 = Device User 2
 */
AHBpmConfigSetting setting = new AHBpmConfigSetting(AHBpmConfig.SwitchUser,1);
```

5.5.13 Blood pressure meter voice control

Turn on voice control

```
AHBpmConfigSetting setting = new AHBpmConfigSetting(AHBpmConfig.VoiceControl,true);
```

Turn off voice control

```
AHBpmConfigSetting setting = new AHBpmConfigSetting(AHBpmConfig.VoiceControl,false);
```

5.5.14 Blood pressure meter manual measurement

Start measuring

```
AHBpmConfigSetting setting = new AHBpmConfigSetting(AHBpmConfig.StartMeasuring);
```

Stop measuring

```
AHBpmConfigSetting setting = new AHBpmConfigSetting(AHBpmConfig.StopMeasuring);
```

5.6. Device Data

5.6.1 Thermometer Data

1. AHTempStatus
   doc-zh/com/aojmedical/plugin/ble/data/temp/AHTempStatus.html

2. AHTempData
   doc-zh/com/aojmedical/plugin/ble/data/temp/AHTempData.html

3. AHTempSummary
   doc-zh/com/aojmedical/plugin/ble/data/temp/AHTempSummary.html

4. AHTempModeData
   doc-zh/com/aojmedical/plugin/ble/data/temp/AHTempModeData.html

5. AHTempErrorData
   doc-zh/com/aojmedical/plugin/ble/data/temp/AHTempErrorData.html

5.6.2 Pulse Oximeter Data

1. AHSpO2
   doc-zh/com/aojmedical/plugin/ble/data/po/AHSpO2.html

2. AHPlethysmogram
   doc-zh/com/aojmedical/plugin/ble/data/po/AHPlethysmogram.html

5.6.3 Blood Pressure Meter Data

1. AHBpmData
   doc-zh/com/aojmedical/plugin/ble/data/bpm/AHBpmData.html

2. AHBpmProcessData
   doc-zh/com/aojmedical/plugin/ble/data/bpm/AHBpmProcessData.html

3. AHBpmConfigData
   doc-zh/com/aojmedical/plugin/ble/data/bpm/AHBpmConfigData.html

4. AHBpmStatus
   doc-zh/com/aojmedical/plugin/ble/data/bpm/AHBpmStatus.html