

VISPEK安卓SDK开发文档

接入步骤

1. 下载`sdklibrary-release.aar`包，放入项目app/libs目录下
2. 在项目的`build.gradle`文件中加入依赖 `implementation files('libs/sdklibrary-release.aar')`,sync一下build.gradle文件
3. 在项目`AndroidManifest.xml`文件中声明蓝牙权限

```
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTHADMIN" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
```

4. 新建一个**MyApplication**类继承Application,在MyApplication类的**OnCreate()**方法中初始化BluetoothClient

```
public BluetoothClient initClient(Context context) {

    if (mClient == null) {
        mClient = new BluetoothClient(context);
    }
    return mClient;
}
```

5. 参考**VispekSDKDemo**进行按需求二次开发，发送指令主要用到**SendCodeUtils**类，调用相应方法发送指令，接收指令主要用到**CodeResponse**类的回调，获取蓝牙返回数值

SDK代码说明

- 搜索蓝牙，需先动态申请权限，蓝牙权限，位置权限必不可少，否则搜不到，还有下拉通知栏蓝牙开关不能关闭状态，位置信息不能关闭，开发者可自行判断加提示。搜不到蓝牙一般就是这几个方面原因。

```
private void searchDevice() {
```

```
    SearchRequest request = new SearchRequest.Builder()
        .searchBluetoothLeDevice(5000, 2).build();

    MyApplication.mClient.search(request, mSearchResponse);
```

```
}

private final SearchResponse mSearchResponse = new SearchResponse() {
```

```
@Override
public void onSearchStarted() {

}

@Override
public void onDeviceFounded(SearchResult device) {

}

@Override
public void onSearchStopped() {

}

@Override
public void onSearchCanceled() {
```

```
}
```

```
};
```

- 连接蓝牙

```
private void connectDeviceIfNeeeded(String address){
```

```
    if (MyApplication.mClient.getConnectStatus(address) == STATUS_DEVICE_CONNECTED ){
        Toast.makeText(this,"蓝牙设备已经被其它应用连接，请先断开再连接",Toast.LENGTH_SHORT).show();
        return;
    }
    MyApplication.mClient.registerConnectStatusListener(address,mConnectStatusListener);
    BleConnectOptions options = new BleConnectOptions.Builder()
        .setConnectRetry(3)
        .setConnectTimeout(20000)
        .setServiceDiscoverRetry(3)
        .setServiceDiscoverTimeout(10000)
        .build();

    MyApplication.mClient.connect(address, options, new BleConnectResponse() {
        @Override
        public void onResponse(int code, BleGattProfile profile) {
            BluetoothLog.v(String.format("profile:\n%s", profile));
            if (code == REQUEST_SUCCESS){

            }
        }
    });
}
```

```
}
```

- 蓝牙连接状态回调

```
private final BleConnectStatusListener mConnectStatusListener = new BleConnectStatusListener() { @Override public void
onConnectStatusChanged(String mac, int status) {
```

```
    //声明一个成员变量用于判断蓝牙连接成功与否
    mConnected = (status == STATUS_CONNECTED);
    tvConnect.setText(mConnected?"断开蓝牙"+blueToothName:"连接蓝牙");
    if (mConnected){
        Toast.makeText(MainActivity.this,"蓝牙连接成功",Toast.LENGTH_SHORT).show();
    }else {
        Toast.makeText(MainActivity.this,"蓝牙已断开",Toast.LENGTH_SHORT).show();
    }
    //判断蓝牙连接成功后，注册一个接收回调，用于接收蓝牙返回的数值
    if (mConnected){
        CodeResponse codeResponse = new CodeResponse(MyApplication.mClient,blueToothAddress);
        codeResponse.registCodeResponseListener((mark, data) -> {
            switch (mark){
                case CodeResponse.CODE_RESPONSE_DEVICE_NO:
                    //设备编号
                    break;
                case CodeResponse.CODE_RESPONSE_ELEC:
                    //电量
                    break;
                case CodeResponse.CODE_RESPONSE_TEMPERATURE:
                    //温度
                    break;
                case CodeResponse.CODE_RESPONSE_VERSION:
                    //硬件版本号
                    break;
                case CodeResponse.CODE_RESPONSE_SCAN:
                    //真实的光谱值 a,a,a,a... 22个
                    break;
                case CodeResponse.CODE_RESPONSE_SCAN_REFERENCE:
                    //真实的参比值 b,b,b,b... 22个
                    break;
            }
        });
    }
}
```

```

        case CodeResponse.CODE_RESPONSE_SCAN_ONE_BY_ONE:
            //发送扫描指令一个灯一个灯的发（基本用不到）
            break;
    }
    /***需要注意点：1.若用户自己搭建算法平台不调用威视佰科后台接口，若设备是22位只需真实的22个光谱值即可，若设备是44位的需要将真实的光谱值和
        2.若用户调用威视佰科后台接口，获取检测值，则忽略第1条。给后台传光谱数据拼接格式为：真实光谱值后面追加参比值即可，不需区分
    });
}
}

```

```
};
```

- 发送指令

```
private void sendCodeIfNeeded(byte[] code){
```

```

    if (MyApplication.mClient!= null && !TextUtils.isEmpty(blueToothAddress)){
        SendCodeUtils.sendCode(MyApplication.mClient, blueToothAddress,code, bleWriteResponse);
    }
}

```

```
}
```

```
//发送指令回调
```

```
private BleWriteResponse bleWriteResponse = new BleWriteResponse() {
```

```

    @Override
    public void onResponse(int code) {
        if (code == REQUEST_SUCCESS){
            Toast.makeText(MainActivity.this,"指令发送成功",Toast.LENGTH_SHORT).show();
        }else {
            Toast.makeText(MainActivity.this,"指令发送失败",Toast.LENGTH_SHORT).show();
        }
    }
}

```

```
};
```