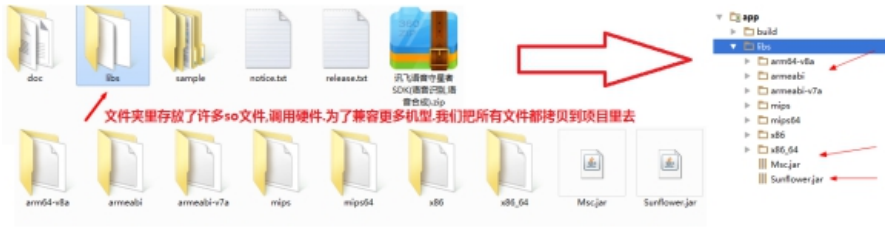
讯飞语音合成

1.0 导入SDK:

将开发工具包中libs目录下的Msc.jar和armeabi(实际还有多个文件,我们全部拷贝进来,以便机型适配)复制到Android工程的libs目录



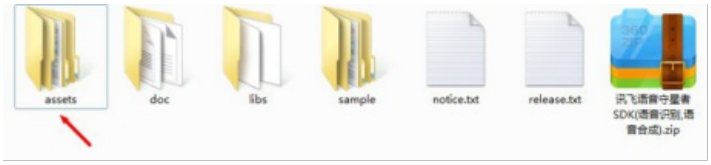
2.0 在项目build.gradle文件的android{}内,配置下面一段代码,同步一下gradle文件.

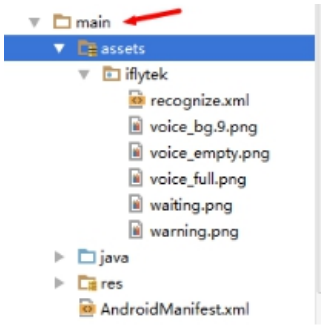
```
sourceSets {
    main {
        jniLibs.srcDir 'libs'
    }
}
```

3.0 对清单文件进行权限配置.

```
<!--连接网络权限，用于执行云端语音能力 -->
<uses-permission android:name="android.permission.INTERNET"/>
<!--获取手机录音机使用权限，听写、识别、语义理解需要用到此权限 -->
<uses-permission android:name="android.permission.RECORD_AUDIO"/>
<!--读取网络信息状态 -->
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<!--获取当前wifi状态 -->
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<!--允许程序改变网络连接状态 -->
<uses-permission android:name="android.permission.CHANGE_NETWORK_STATE"/>
<!--读取手机信息权限 -->
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
<!--读取联系人权限，上传联系人需要用到此权限 -->
<uses-permission android:name="android.permission.READ_CONTACTS"/>
<!-- 如需使用人脸识别，还要添加：摄相头权限，拍照需要用到 -->
<uses-permission android:name="android.permission.CAMERA" />
```

4.0 为了便于快速开发,SDK提供了一套默认的语音输入UI,若使用,请将SDK资源包assets下的文件拷贝至项目的asstes目录下(studio没有,就把assets复制到src->main下)





5.0 初始化讯飞语音SDK:

只有初始化后才可以使用MSC的各项服务。建议将初始化放在程序入口处（如Application、Activity的onCreate方法),初始化代码如下：

```
public class APP extends Application {
@Override
public void onCreate() {
super.onCreate();
// 将“12345678”替换成您申请的APPID，申请地址：http://open.voicecloud.cn
SpeechUtility.createUtility(this, SpeechConstant.APPID +"=58189441");
}
}
```

注意:自定义的APPlication要到清单文件配置一下.在application的">"内,填上android:name=".APP"

6.0 创建Bean类(名字自定义,如果使用本文章全套代码,那么就暂时不要改,牵一发动全身),以用来存放讯飞语音识别的结果json数据.

```
public class XFBean {
    public ArrayList<WS> ws;
    public class WS{
        public ArrayList<CW> cw;
    }
    public class CW{
        public String w;
    }
}
```

7.0 下面是mainActivity中的代码块

执行识别语音为文字的的逻辑代码(bean解析出来的最终String类型数据)

```
public void Listen(View view) {
        //1.创建RecognizerDialog对象,第二个参数就是一个初始化的监听器,我们用不上就设置
```

```
为null
        RecognizerDialog mDialog = new RecognizerDialog(this, null);
        //2.设置accent、language等参数
        mDialog.setParameter(SpeechConstant.LANGUAGE, "zh_cn");//设置为中文模式
        mDialog.setParameter(SpeechConstant.ACCENT, "mandarin");//设置普通话模式
        //若要将UI控件用于语义理解，必须添加以下参数设置，设置之后onResult回调返回将是
语义理解
        //mDialog.setParameter("asr_sch", "1");
        //mDialog.setParameter("nlp_version", "2.0");
        //创建一个装每次解析数据的容器
        mStringBuilder = new StringBuilder();
        //3.设置回调接口
        mDialog.setListener(new RecognizerDialogListener() {
            @Override//识别成功执行,参数recognizerResult 识别的结果,Json格式的字符串
            //第二参数 b:等于true时会话结束.方法才不会继续回调
            //一般情况下通过onResult接口多次返回结果,完整识别内容是多次结果累加的
            public void onResult(RecognizerResult recognizerResult, boolean b) {
                //拿到讯飞识别的结果
                String resultString = recognizerResult.getResultString();
/*                System.out.println("讯飞识别的结果 " + resultString);
                System.out.println("b参数是什么 " + b);*/
                //自定义解析bean数据的方法,得到解析数据
                String content= parseData(resultString);
//                System.out.println("解析后的数据 "+ content);
                mStringBuilder.append(content);
                //对参数2b进行判断,如果为true,代表这个方法不会对调,那么我们容器的数据转
为字符串,拿来使用即可
                if(b){
                    String result = mStringBuilder.toString();
                    System.out.println(result);
                    //回答对象,在没有匹配到用户说的话,默认输出语句
                    String anwser="不好意思,年纪大了,耳朵不好,没有听清楚";
                    if(result.contains("你好")){
                        anwser="你好,我是你的智能语音助手,很高兴为你服务";
                    }else if(result.contains("安卓学习哪家强")){
                        anwser="快到北京,找至远教育";
                    }else if(result.contains("美女")){
                        String [] anwserList=new String[]{"你是坏人不和你玩","小助手
很纯洁,不要说我听不懂的话","小助手我就是美女,主人","500元,小助手帮主人找美女一起打英雄联
盟"};
                        int random = (int)(Math.random() * anwserList.length);
                        anwser=anwserList[random];
                    }
                    shuo(anwser);
```

```
            }

        }

        @Override//识别失败执行的方法,speechError错误码
        public void onError(SpeechError speechError) {
            System.out.println("错误码 " + speechError);
        }
    });
    //4.显示dialog，接收语音输入
    mDialog.show();
}

/**
  * 把文字转换为声音
  */
public void Talk(View view) {
    shuo("至远教育,让你月薪过万不是梦");
}

public void shuo(String result){
    //1.创建 SpeechSynthesizer 对象, 第二个参数：本地合成时传 InitListener
    SpeechSynthesizer mTts= SpeechSynthesizer.createSynthesizer(this, null);
    //2.合成参数设置，详见《MSC Reference Manual》SpeechSynthesizer 类
    // 设置发音人（更多在线发音人，用户可参见 附录13.2
    mTts.setParameter(SpeechConstant.VOICE_NAME, "xiaoyan"); //设置发音人
    mTts.setParameter(SpeechConstant.SPEED, "50");//设置语速
    mTts.setParameter(SpeechConstant.VOLUME, "80");//设置音量，范围 0~100
    mTts.setParameter(SpeechConstant.ENGINE_TYPE, SpeechConstant.TYPE_CLOUD); //
设置云端,这些功能用到了讯飞服务器,所以要有网络
    //设置合成音频保存位置（可自定义保存位置），保存在“./sdcard/iflytek.pcm”
    //保存在 SD 卡需要在 AndroidManifest.xml 添加写 SD 卡权限
    //仅支持保存为 pcm 和 wav 格式，如果不需要保存合成音频，注释该行代码
    // mTts.setParameter(SpeechConstant.TTS_AUDIO_PATH, "./sdcard/iflytek.pcm");
    // 3.开始合成,第一个参数就是转换成声音的文字,自定义,第二个参数就是合成监听器对象,我们不需
要对声音有什么特殊处理,就传null
    mTts.startSpeaking(result, null);
}
```

创建解析讯飞识别结果bean数据的方法,使用谷歌的Gson.

```
private String parseData(String resultString){
    //创建gson对象.记得要关联一下gson.jar包,方可以使用
    Gson gson = new Gson();
    //参数1 String类型的json数据    参数2.存放json数据对应的bean类
```

```
        XFBean xfBean = gson.fromJson(resultString, XFBean.class);
        //创建集合,用来存放bean类里的对象
        ArrayList<XFBean.WS> ws=xfBean.ws;
        //创建一个容器,用来存放从每个集合里拿到的数据,使用StringBUndle效率高
        StringBuilder stringBuilder = new StringBuilder();
        for (XFBean.WS w : ws) {
            String text= w.cw.get(0).w;
            stringBuilder.append(text);
        }
        //把容器内的数据转换为字符串返回出去
        return stringBuilder.toString();
    }
```

执行识别语音为文字的的逻辑代码.(没使用bean包的代码)

```
//1.创建RecognizerDialog对象,第二参数就是一个初始化的监听,我们用不上就设置为null
RecognizerDialog mDialog = new RecognizerDialog(this,null);
//2.设置accent、language等参数
mDialog.setParameter(SpeechConstant.LANGUAGE, "zh_cn");//设置为中文模式
mDialog.setParameter(SpeechConstant.ACCENT, "mandarin");//设置为普通话模式
//若要将UI控件用于语义理解，必须添加以下参数设置，设置之后onResult回调返回将是语义理解
//mDialog.setParameter("asr_sch", "1");
//mDialog.setParameter("nlp_version", "2.0");
//3.设置回调接口,语音识别后,得到数据,做响应的处理.
mDialog.setListener(new RecognizerDialogListener() {
//识别成功执行 参数1 recognizerResult:识别出的结果,Json格式(用户可参见附录12.1)
// 参数2 b:等于true时会话结束。方法才不会继续回调
//一般情况下会通过onResults接口多次返回结果，完整的识别内容是多次结果的累加,(关于解析Json的代码可参见MscDemo中JsonParser类)
@Override
public void onResult(RecognizerResult recognizerResult, boolean b) {
//拿到讯飞识别的结果
String resultString = recognizerResult.getResultString();
System.out.println("讯飞识别的结果 "+resultString);
System.out.println("b参数是什么 "+b);
}
@Override//识别失败执行 speechError:错误码
public void onError(SpeechError speechError) {
}
});
//4.显示dialog，接收语音输入
mDialog.show();
```