**智能机器人实训报告**

专业班级： 姓名： 学号：

所属学院： 指导教师：

一、实验目标

本次实验通过AIUI的配置及调用，开发一个关于语音识别的场景应用，了解语音识别技术如何在机器人上使用。本实验的具体实验目标如下：

1. 了解AIUI开放平台账号申请、应用创建及配置；
2. 掌握使用开发套件完成机器人智能应用的开发与调试；
3. 掌握语音识别能力接口的调用开发。
4. 实验内容

语音交互技术作为人工智能领域的核心组成部分，也是智能机器人技术发展的重要要素。语音交互技术主要包含以下几个方面：

* 1. 语音合成技术，指的是涉及声学、语言学、数字信号处理、计算机科学等多个学科技术，是中文信息处理领域的一项前沿技术，解决的主要问题是如何将文字信息转化为可听的声音信息，也即“让机器像人一样开口说话”；
  2. 语音识别技术，指的是所要解决的问题是让机器能够“听懂”人类的语音，将语音中包含的文字信息“提取”出来，相当于给机器安装上“耳朵”，使其具备“能听”的功能；
  3. 语义理解技术，指的是将一句自然语言转化为计算机可读的结构化数据。语义理解也是AIUI 的核心功能。例如“明天上海天气怎么样”这句自然语言中，经过AIUI语义理解引擎，得到这句话所对应具有固定格式的Json数据。

本次实验通过开发一个实际的简单智能应用，来对机器人使用的语音识别技术使用方法做详细的讲解，实验的具体流程如下：

# 实验步骤

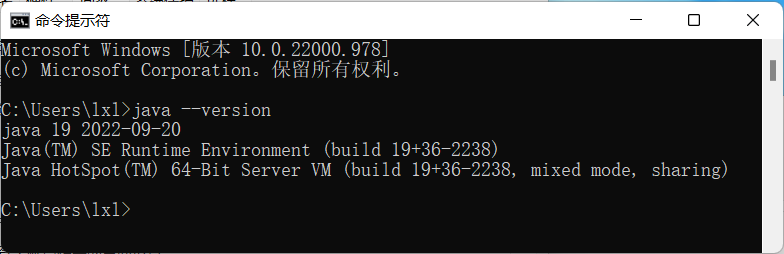
**步骤1：****AIUI账号申请及平台设置**

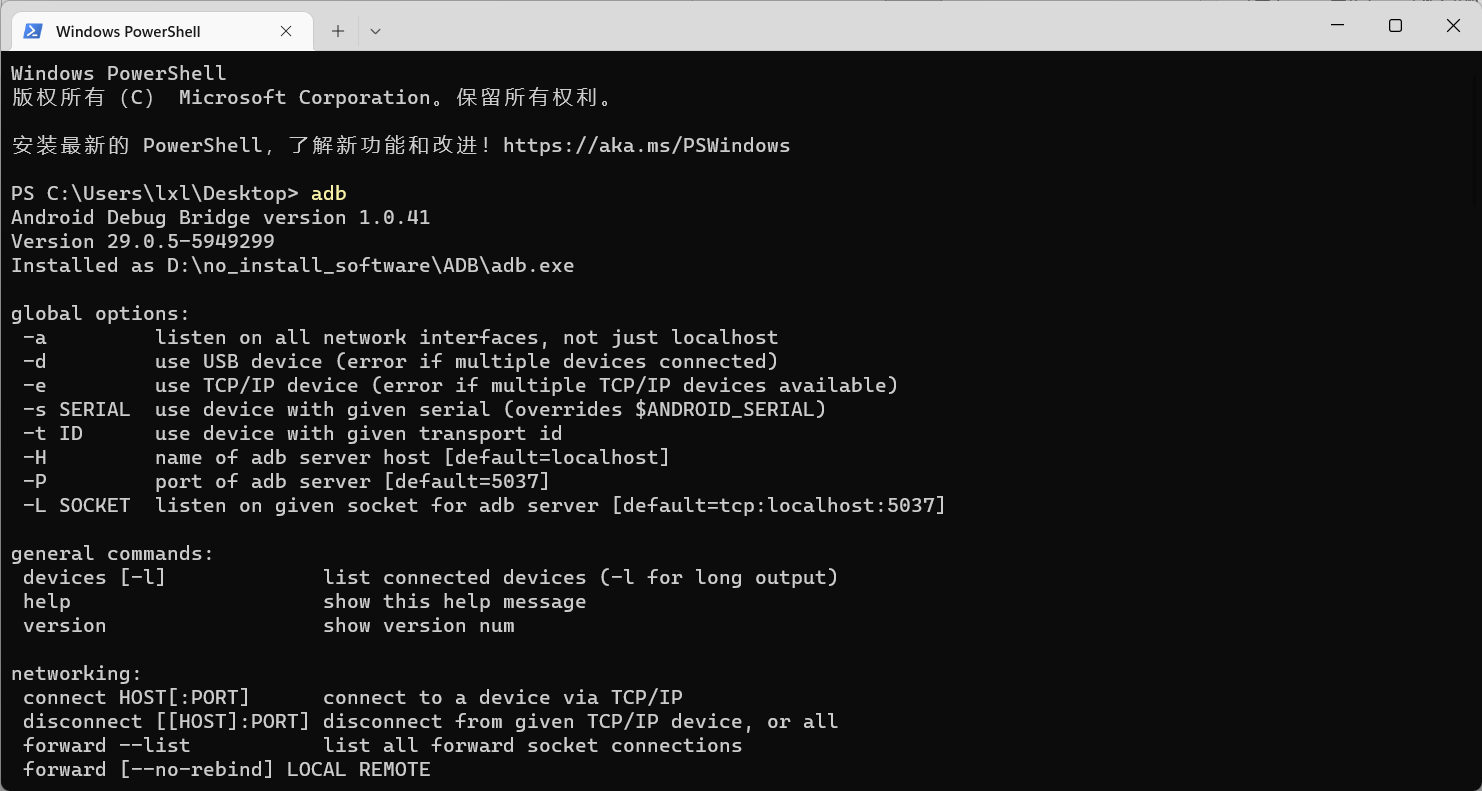
由于本次实验开发语音合成功能应用使用的是科大讯飞的AIUI语音合成功能，我们需要先到讯飞的AIUI平台申请账号，并完成平台上相关的功能设置。你申请的账号情况：

用户名：18259734087 APPID: 7d6c0cb2 APPKEY: 0843932be993851f4b45a21ed7381f3e 。

**步骤2：****项目工程搭建及能力包导入**

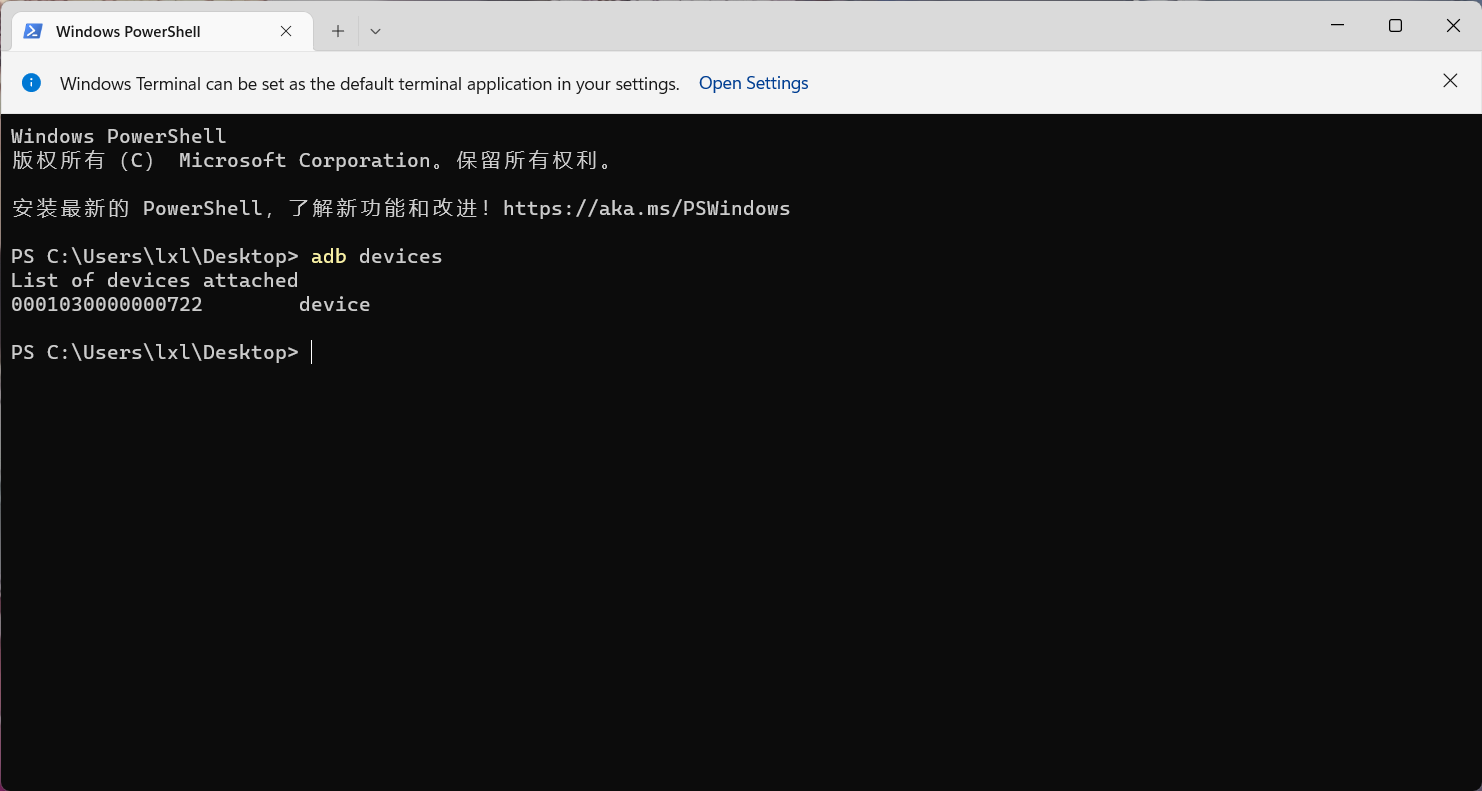
机器人使用的是Android平台，所有机器人的智能应用都是基于Android环境运行，机器人应用开发跟普通的Android应用开发有很大类似，在进行机器人智能应用开发之前，我们需要先在开发电脑上搭建安卓的开发环境，包括运行的JDK的安装及配置、开发工具的安装及配置等。实验的具体流程如下：

你下载的JDK版本为：

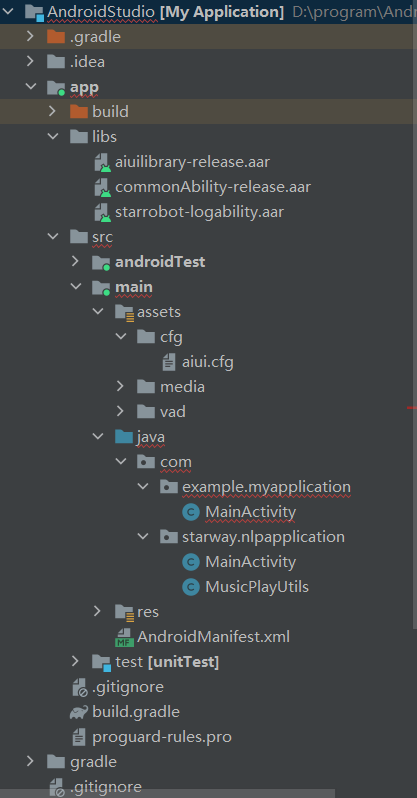
ADB工具环境配置结果为：

Android Studio版本为：\_\_\_android-studio-2021.3.1.16\_\_\_\_\_\_\_\_\_

安装时设置的SDK目录为：\_\_\_D:\software\AndroidStudio\_\_\_\_\_\_\_\_\_\_

我们使用USB线连接机器人开发平台与电脑，完成连接后，我们通过ADB工具查看机器人开发平台与电脑的连接状态为：。

完成了项目工程的创建后，我们现在开始导入机器人开发的能力包，首先，我们导入应用必需的机器人基础能力开发包和需要依赖的日志能力包，以及AIUI能力开发包，完成后项目整体目录结构为（**截图**）：



完成了应用的能力包导入后，我们需要对工程的能力包依赖关系及权限配置等信息进行配置，主要流程如下：

1. 在build.gradle文件中添加能力包的依赖；

|  |
| --- |
| buildscript {  repositories {  google()  jcenter()  }  dependencies {  classpath "com.android.tools.build:gradle:4.1.1"  // NOTE: Do not place your application dependencies here; they belong  // in the individual module build.gradle files  }  }  allprojects {  repositories {  google()  jcenter()  }  }  task clean(type: Delete) {  delete rootProject.buildDir  } |

1. 在AndroidManifest.xml文件中添加应用权限设置；

|  |
| --- |
| *<?*xml version="1.0" encoding="utf-8"*?>* <manifest xmlns:android="http://schemas.android.com/apk/res/android"  package="com.starway.myapplication">  *<!--六麦唤醒-->* <uses-permission android:name="com.android.permission.IFLYTEK\_MIC\_WAKEUP" />  *<!-- 读取内存卡权限 -->* <uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE"/>  *<!-- 连接网络权限 -->* <uses-permission android:name="android.permission.INTERNET"/>  *<!-- 允许程序录制音频 -->* <uses-permission android:name="android.permission.RECORD\_AUDIO"/>  *<!-- 允许程序访问有关的网络信息 -->* <uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE"/>  *<!-- 允许程序改变网络连接状态 -->* <uses-permission android:name="android.permission.CHANGE\_NETWORK\_STATE"/>  *<!-- 允许程序访问Wi-Fi网络状态信息 -->* <uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE"/>  <uses-permission android:name="android.permission.CHANGE\_WIFI\_STATE"/>  *<!-- 允许应用写（非读）用户的外部存储器 -->* <uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE"/>  *<!-- 允许程序读取所有者数据 -->* <uses-permission android:name="android.permission.READ\_OWNER\_DATA"/>  *<!-- 允许一个程序写入但不读取所有者数据 -->* <uses-permission android:name="android.permission.WRITE\_OWNER\_DATA"/>   <application  android:name=".DemoApplication"  android:allowBackup="true"  android:icon="@mipmap/ic\_launcher"  android:label="@string/app\_name"  android:roundIcon="@mipmap/ic\_launcher\_round"  android:supportsRtl="true"  android:theme="@style/AppTheme">  <activity android:name=".MainActivity"  android:launchMode="singleTask">  <intent-filter>  <action android:name="android.intent.action.MAIN"/>  <category android:name="android.intent.category.HOME" />  <category android:name="android.intent.category.DEFAULT"/>  <category android:name="android.intent.category.LAUNCHER"/>  </intent-filter>  </activity>  </application>  </manifest> |

1. 在项目工程的MainActivity类中添加机器人的能力初始化方法，并在场景初始化的onCreate方法中添加调用。

请将上述三个流程涉及的**程序代码**复制在下方：

|  |
| --- |
| package com.example.myapplication;  import androidx.appcompat.app.AppCompatActivity;  import android.os.Bundle;  import android.os.Handler;  import com.starway.starrobot.commonability.RobotType;  import com.starway.starrobot.commonability.StarCommonAbility;  import com.starway.starrobot.commonability.hardware.EmojiHelper;  import com.starway.starrobot.logability.StarLogAbility;  public class MainActivity extends AppCompatActivity {  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);  initRobotAbility();  }  private void initRobotAbility() {  //日志初始化  StarLogAbility.getInstance().initAbility(this);  StarCommonAbility.getInstance().initAbility(this.getApplicationContext(), RobotType.TYPE\_TEACHING, new StarCommonAbility.onResultCallback() {  @Override  public void onResult(boolean isSuccess, String hard\_code) {  if (isSuccess) {  //硬件和业务状态初始化  switch (hard\_code) {  case "emoji":  EmojiHelper.doEmojiBase();  showEmojiEffectDelay();  break;  default:  break;  }  }  }  });  }  private void showEmojiEffectDelay()  {  new Handler().postDelayed(new Runnable() {  @Override  public void run() {  EmojiHelper.*doEmojiLove*();  }  },3000);  }  } |