Voice Recognition

Android comes with a built-in feature speech to text through which you can provide speech input to your app. With this you can add some of the cool features to your app like adding voice navigation(Helpful when you are targeting disabled people), filling a form with voice input etc.,

In the background how voice input works is, the speech input will be streamed to a server, on the server voice will be converted to text and finally text will be sent back to our app.

Here, we will create a simple app to demonstrate this tutorial. This app contains a simple button to invoke speech input and a TextView to display the converted speech text.

1. Create a new project in android Studio by going to **File ⇒ New ⇒ Android Application Project** and give required information.

   ```
   Application Name: ttsdemo
   Company Domain: com.android.ttsdemo
   ```

2. Open colors.xml (or new added) located under res ⇒ values and add below colors. If you don’t see colors.xml, create a new file and add the values.

   ```
   <?xml version="1.0" encoding="utf-8"?>
   <resources>
     <color name="white">#ffffff</color>
     <color name="bg_gradient_start">#31244e</color>
     <color name="bg_gradient_end">#6b394c</color>
   </resources>
   ```

3. Now open the layout file for "activity_main.xml" and add below code to create a simple layout.

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/bg_gradient"
    android:orientation="vertical" >

    <TextView
        android:id="@+id/txtSpeechInput"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="100dp"
        android:textColor="@color/white"
        android:textSize="26dp"
        android:textStyle="normal" />

    <Button
        android:id="@+id/btnSpeak"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Speaking" />

</RelativeLayout>
```

- Now the core of App, MainActivity.java: there are two steps to implement speech input:
- Starting RecognizerIntent
  First we need to create a RecognizerIntent by setting necessary flags such as

  ```
     ACTION_RECOGNIZE_SPEECH – Simply takes user’s speech input and retur
  ns it to same activity
     LANGUAGE_MODEL_FREE_FORM – Considers input in free form English
     EXTRA_PROMPT – Text prompt to show to the user when asking them to s
  peak
  ```

- Step 2: Receiving the speech response
  Once the speech input is done we have to catch the response in onActivityResult and take appropriate action needed.

MainActivity.java

```
package tts.android.com.ttsdemo;

import java.util.ArrayList;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.speech.RecognizerIntent;
```

```
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class MainActivity extends Activity {

    private TextView txtSpeechInput;
    private Button btnSpeak;
    //private final int REQ_CODE_SPEECH_INPUT = 100;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        txtSpeechInput = (TextView) findViewById(R.id.txtSpeechInput);
        btnSpeak = (Button) findViewById(R.id.btnSpeak);

        // hide the action bar
        //getActionBar().hide();

        btnSpeak.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //open buld-in Voice Recognition Activity by Intent ...
                Intent intent = new Intent(RecognizerIntent.ACTION_RECOGNI
ZE_SPEECH);
                intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL, Rec
ognizerIntent.LANGUAGE_MODEL_FREE_FORM);
                // show the prompt
                intent.putExtra(RecognizerIntent.EXTRA_PROMPT, "Listening.
..");

                startActivityForResult(intent, 1);
            }
        });
    }

    /**
     * Receiving speech input
     * */
    @Override

    protected void onActivityResult(int requestCode, int resultCode, Inten
t data ) {
        super.onActivityResult(requestCode, resultCode, data);

        if (requestCode == 1) {
            if (resultCode == RESULT_OK && null != data) {

                ArrayList<String> result = data.getStringArrayListExtra(Re
cognizerIntent.EXTRA_RESULTS);
                txtSpeechInput.setText(result.get(0));
            }
        }
    }

}
```

After finished, make sure that the device has good internet connectivity while you are testing.

Note

ANDROID STUDIO EMULATOR HAXM ON MAC OS HIGH SERRIA (10.13)

Update to android Studio, it suggests to upgrade last HAXM.