

WEB ICP11

DHARMA TEJA K

Email: dkbmy@umsystem.edu

GitHub Link: <https://github.com/Dharmateja183/Web-and-Mobile-programming-spring-2022/tree/main/mobile%20part/ICP11>

Avinash Reddy T

Email: atfkh@umsystem.edu

GitHub link: <https://github.com/avinashreddy3/WebDevCourse/tree/main/MobilePart/ICP11>

In Class Programming:

In this ICP, we are going to utilize android studio, java programming, xml languages for mobile application development,

To create an app with Text-to-Speech functionality which convert the text that is entered on the field into speech using android studio.

Procedure:

First we have created a new project with empty activity using android studio application.

This provides .java, .xml files which are used to convert the text entered by the user into speech.

1. Creating a TextView, button & EditText fields in android manifest.xml file.

We've added TextView, EditText, and Button fields to the 'activity main.xml' file to collect user input and transform it to voice.

```
<?xml version="1.0" encoding="utf-8"?>

<!--relative layout is view group which shows the child view in their relative positions-->
<!-- here we added top padding to our android app-->
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingTop="65dp"
    tools:context=".MainActivity">
```

To all the fields, we have added height, weight, color, alignment properties and all ,as show in below snippets.

EditView:

To accept the input text from the user to convert it into speech

```
<!-- added one edittext field and created an id -->
<EditText
    android:id="@+id/editTextTTS"
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:hint="enter your text"
    android:layout_below="@+id/textViewTTS"
    android:layout_centerHorizontal="true"
    android:layout_margin="35dp"
/>
```

TextView:

This acts like main heading or title of our android application

```

<!--    added one text view feature with below parameters-->
<TextView
    android:id="@+id/textViewTTS"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Text to Speech"
    android:textColor="#ff1c1c"
    android:textSize="35dp"
    android:layout_centerHorizontal="true"
/>

```

Button:

The button field assisted in the conversion of written text to voice. When the "Speak" button is triggered, the text entered is converted into speech.

```

<!--    added a user interactive speak button -->
<Button
    android:id="@+id/btnTTS"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/editTextTTS"
    android:layout_centerHorizontal="true"
    android:text="Speak"
/>
</RelativeLayout>

```

2. (Adding Text-to-Speech capabilities to turn text into speech): MainActivity.java:

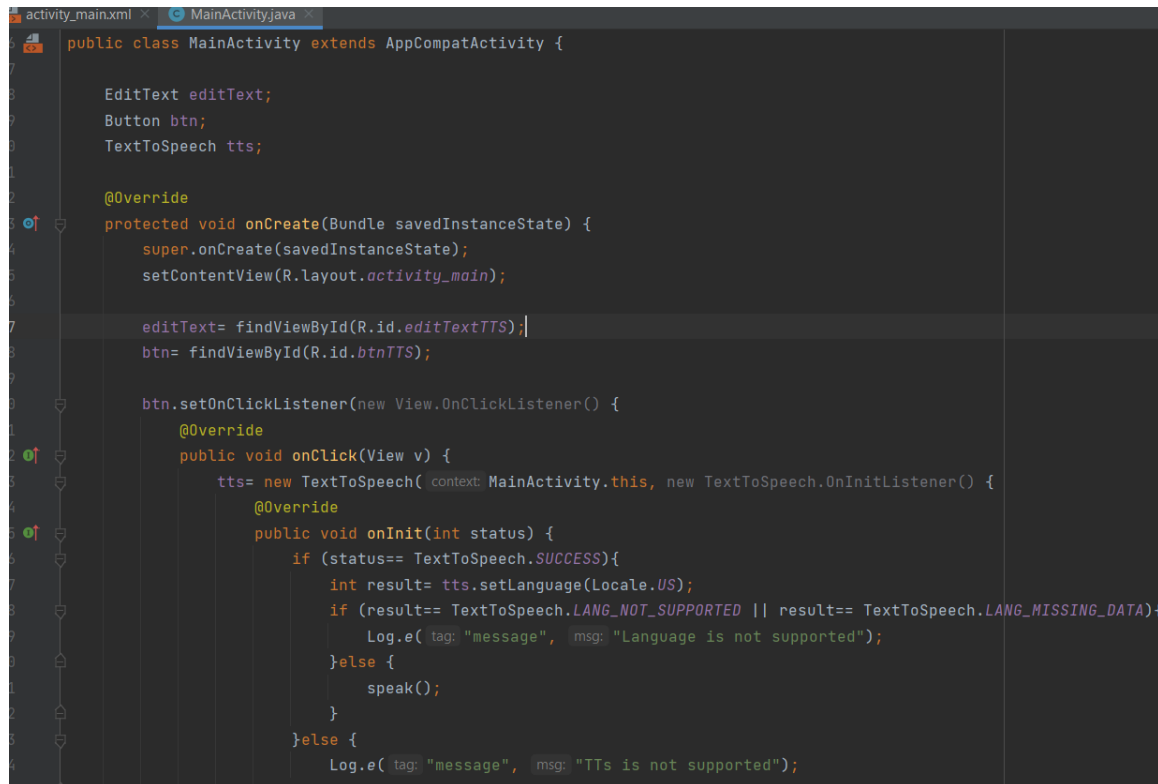
First, we had given the id's for each of the fields we created above.

And create the variables for the three text **Text-to-Speech** functionalities which are EditText, Button, TextToSpeech objects as shown below:

After we've finished creating the object, we've implemented validations for both success and failure scenarios.

If no errors are identified, it will execute properly and pick "Locale.US" as the language of speech. Validations for language selection have also been included.

If the specified language is incorrect or not functioning correctly, an error notice will appear. Otherwise, the text will be successfully transformed. This is shown in below screenshot.



```
public class MainActivity extends AppCompatActivity {

    EditText editText;
    Button btn;
    TextToSpeech tts;

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

        editText= findViewById(R.id.editTextTTS);
        btn= findViewById(R.id.btnTTS);

        btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                tts= new TextToSpeech( context: MainActivity.this, new TextToSpeech.OnInitListener() {
                    @Override
                    public void onInit(int status) {
                        if (status== TextToSpeech.SUCCESS){
                            int result= tts.setLanguage(Locale.US);
                            if (result== TextToSpeech.LANG_NOT_SUPPORTED || result== TextToSpeech.LANG_MISSING_DATA){
                                Log.e( tag: "message", msg: "Language is not supported");
                            }else {
                                speak();
                            }
                        }else {
                            Log.e( tag: "message", msg: "TTS is not supported");
                        }
                    }
                });
            }
        });
    }
}
```

And the next main feature is speak() method/ function which is to be triggered when clicked on the speak button on the home page.

Here we have also added the setSpeechRate() which is used to control the rate of speed on voice.

And we have given the two scenarios here as well,

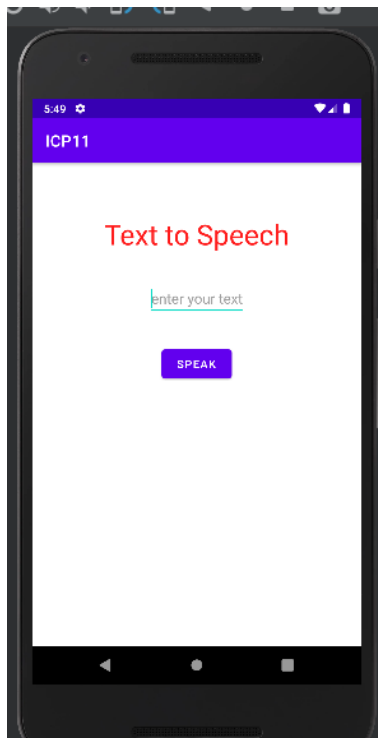
- If the text entered is empty, display a message to user to enter the text to speak
- Else convert the text entered to voice.

The "stop()" method was also used to interrupt the current sentence (whether it was being played or stored to a file) and discard the remaining sentences in the queue.

We also utilized the shutdown() method, which is intended to free up native TextToSpeech resources.

```
void speak() {  
    String s = String.valueOf(editText.getText());  
    tts.setSpeechRate(0.1f);  
    if (isEmpty(s)){  
        tts.speak( text: "please enter something to speak", TextToSpeech.QUEUE_ADD, params: null);  
    }else{  
        tts.speak(s, TextToSpeech.QUEUE_ADD, params: null);  
    }  
}  
  
@Override  
protected void onDestroy(){  
    super.onDestroy();  
    tts.stop();  
    tts.shutdown();  
}
```

Output pages on the android device:



We have entered the text to speech, as shown below, to test:

