Audio Recording

Audio Recording in Android

Two options

- AudioRecorder
  - Produces raw audio files (no compressing, encoding/decoding)
  - Raw audio files are not automatically recognized by standard media player
- MediaRecorder
  - Produces formatted audio files: .M4A, .3GPP, .AAC, .OGG

Step 1: Add Dependencies (Media v3)

```
dependencies {
    // Media3
    implementation("androidx.media3:media3-exoplayer:1.8.0")
    implementation("androidx.media3:media3-common-ktx:1.8.0")
    implementation("androidx.media3:media3-session:1.8.0")

    // For permission handling at runtime
    implementation("com.google.accompanist:accompanist-permissions:0.37.3")


}
```


Step 2: Add Permissions (AndroidManifest.xml)

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

  <uses-permission android:name="android.permission.RECORD_AUDIO "/>
  <application android:allowBackup="true">
    <activity android:name=".MainActivity">
      <intent-filter>
        <action android:name="androidx.intent.action.MAIN" />"
        <category android:name="androidx.intent.category.LAUNCHER" />"
      </intent-filter>
    </activity>
  </application>
</manifest>
```

Step 4a: Create MediaRecorder (in ViewModel)

```
class AppViewModel(val app: Application): AndroidViewMode(app) {
  private var recorder: MediaRecorder? = null
  private audioFile: String = “”

  private fun createRecorder() = MediaRecorder(app.applicationContenxt).apply {
      setAudioSource(MediaRecorder.AudioSource.MIC)
      setOutputFormat(MediaRecorder.OutputFormat.THREE_GPP)
      val auPath = app.applicationContext.externalCacheDir?.absolutePath
      audioFile = “$auPath/audio_rec.3gpp”
      setOutputFile(audioFile)
      setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB)
  }
}
```

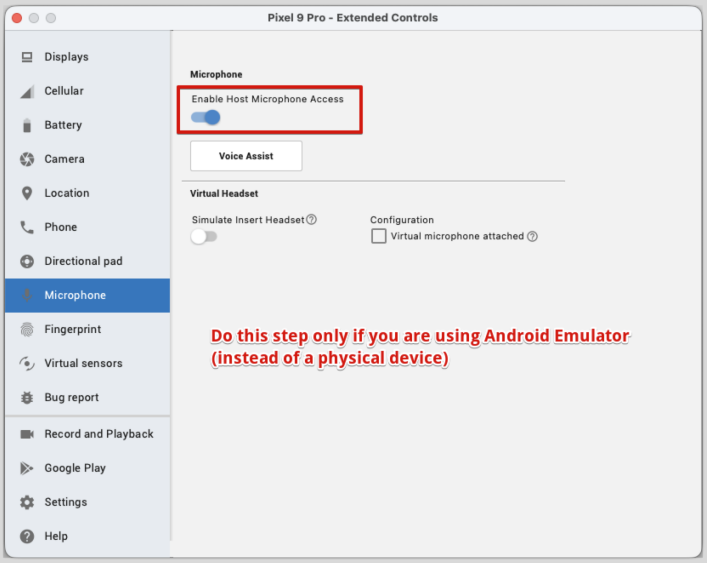

Step 4b: Start/Stop Recorder

```
class AppViewModel(val app: Application): AndroidViewMode(app) {
  private var recorder: MediaRecorder? = null
  fun startRecording() {
      try {
          if (recorder == null) recorder = createRecorder()
          recorder?.prepare()
          recorder?.start()
     } catch(e: IOException)
         println("Can't initialize MediaRecorder ${e.message}")
  }

  fun stopRecording() {
      recorder?.stop()
      recorder?.release()
  }
}
```

Step 4c: Play Recorder Audio

```
class AppViewModel(val app: Application): AndroidViewMode(app) {
    // See details of MediaController initialization in the Audio Service slides
    private var mediaController: MediaController
    private var audioFile: String = “”   // initialized in createRecorder()

    fun playRecording() {
        val audioUri = Uri.encode(audioFile)
        val mediaItem = MediaItem.fromUri(audioUri)
        mediaController.repeatMode = Player.REPEAT_MODE_OFF
        mediaController?.setMediaItem(mediaItem)
        mediaController?.prepare()
        mediaController?.play()
    }
}
```


Connect Emulator Microphone



@Composable: Request Permission

```
@Composable AudioScreen(viewModel:  AppViewModel) {
    val audioRecordPermission = rememberPermissionState(Manifest.permission.RECORD_AUDIO)

    LaunchedEffect(isPlayingAudio) {
        if (!audioRecordPermission.status.isGranted)
            audioRecordPermission.launchPermissionRequest()
    }
    Row {
       Button(onClick = { vm.startRecording() }) { Text("Record") }
       Button(onClick = { vm.stopRecording()  }) { Text("Stop")   }
       Button(onClick = { vm.playRecording()  }) { Text("Play")   }
    }
}
```