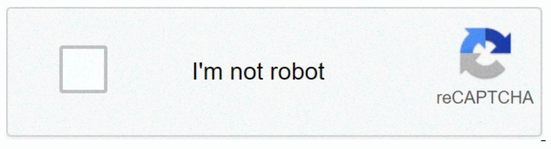

Continue

Audio recorder android tutorial

There are excellent apps on the Android Play Store that use voice recording features from your mobile device, some great examples include apps such as Shazam, which listens to a song playing to discover the name of the song and artist, as well as Otter, the app that listens and takes meeting notes of your verbal conversations. I did some research and created the very basic voice recorder app in Android and published the source code at GitHub. I'll share with you in this article how to record programmatically audio in Android. Android offers a MediaRecorder class to record audio from the built-in microphone on your Android device. In order to make use of the MediaRecorder class to record audio programmatically in Android, follow the steps below. Step Description 1Eee permission to record audio in manifest file 2Seee permission to record audio when creating Activity3Add UI components such as buttons to trigger the voice recording4Use the MediaRecorder class to start a voice recording when the user selects a button to start recording5Use the MediaRecorder class to stop voice recording when the user selects a button to stop recording in the next section , I'll go into detail on the features offered by the MediaRecorder class. From there, I'll take you through the step-by-step process for audio recording in Android. MediaRecorder Class Summary Before you can use the MediaRecorder to record audio, you'll need to make sure you've got permission to record audio through the app's manifest file and main activity. I'll cover this in the tutorial below. Assuming you have permission to record audio, in order to start recording, you will first have to import the MediaRecorder class and then create a new MediaRecorder class instance. From there, you'll need to set up your audio recorder by setting the audio source, audio encoder, output format, and output file on your MediaRecorder instance. To start recording, use the preparation and start-up method on the MediaRecorder. Please see a summary of the important methods of the MediaRecorder class. MethodDescriptions andAudioSource (int)This method defines the audio source to be used for recording. The audio source must be set before setting the output format. For the audio source microphone use MediaRecorder.AudioSource.MICOther examples for audio sources can be found in audioSource documentationsetOutputFormat(int)This method defines the format of the output file for Audio. For the 3GPP format, use MediaRecorder.OutputFormat.THREE_GPPOther examples for output formats that you can use are available in the OutputFormat documentation. Please note that the MediaRecorder can be used for video and audio recording.setAudioEncoder(int)The method defines the audio encoder to be used for audio recording. For the use of the AMR audio codec (narrow band), MediaRecorder.AudioEncoder.AMR_NBOther examples for audio coders that you can use are in AudioEncoder documentation.setOutputFile(String)This method defines the trajectory of where the output file will be generated for audio recording.prepare()This method prepares the recorder to capture and encode datastart()This method begins to capture and encode the data from the recorder to the file specified in the setOutputFormat method stop. In order to start recording again, you will need to reconfigure the recorder and call prepare and then start.release()This method will free up the resources associated with the MediaRecorder. Is usually a good practice to call the release after stopping recording with the method.pause stop()The method will pause the recording and allow you to resume using the CV () methodresume ()The method will allow you to resume recording if your recording has been interrupted using the pause () method requesting permission to record audio in Android In order to record the audio in your app, you will need to use the permission android.permission.RECORD_AUDIO. Due to the potential privacy risks to the user during audio recording, the Android team has classified RECORD_AUDIO privacy authorization as a dangerous authorization. Like all permissions, you'll need to ask for permission RECORD_AUDIO in your Android app's manifest file. In addition, because permission RECORD_AUDIO is a dangerous authorization, you will need to obtain explicit consent from users to use that permission. This can be done through a quick application time asking for permission to use RECORD_AUDIO in your app. I'll show you how to do it in an activity in the step-by-step tutorial below. Steps to record audio programmatically in Android The next section of this post will include a step-by-step tutorial to show you how to record audio programmatically in Android. This sample code is available in a public GitHub repository called AudioRecorder. Request permission to record audio in the manifest In order to record audio using the MediaRecorder class, you will need to include permission RECORD_AUDIO in your app's manifest file. Your manifest file can be located at the path/src/main/AndroidManifest app.xml Inside the overt label above the app label, add a use-permission item requesting android.permission.RECORD_AUDIO permission. Requesting permission to record audio in the activity Due to privacy risks, the Android team considers the authorization RECORD_AUDIO is considered a dangerous authorization. With dangerous permissions, you will need to obtain consent user to use these permissions. If you don't, a SecurityException will be launched if the app attempts to use a dangerous authorization that has not received consent. To obtain the user's explicit consent to use the RECORD_AUDIO permission, do the following within the MainActivity class. Set a string chart of permissions that you will ask for consent to use. In this case, we are only in Manifest.permission.RECORD_AUDIODefine a boolean to check if the audio recording authorization has been created, the falseDefine defect a constant whole representing the request for permission to record audioOverride the method onRequestPermissionsResult to verify that permission to record the audio has been granted and update the boolean value accordingly. If permission has not been granted close the application using the finishing method ()In the face of the onCreate method (...) call the method ActivityCompat.requestPermissions (...) to request audio recording permission Adding UI components to the voice recorder application, I will trigger the start and stop of the audio recording via buttons in the user interface. For the mainActivity layout, I'll create a basic vertically oriented LinearLayout and display a button to start the audio recording and a button to stop the audio recording. Below is an excerpt from the file activity_main.xml layout. Inside the MainActivity class in the onCreate method (...), I will call setContentView (R.layout.activity_main); Use the layout file Using the MediaRecorder class to start and stop recording audio to start recording audio using the MediaRecorder perform the following steps: Create a new instance of the MediaRecorderSet class audio source on the MediaRecorder to MediaRecorder. AudioSource.MICSet the output format on the MediaRecorder to MediaRecorder.OutputFormat.THREE_GPPSet the output file path on the MediaRecorderSet encode it audio on the MediaRecorder to MediaRecorder.AudioEncoder.AMR_NBCall the preparation method () on the MediaRecorderCall method at the beginning() on the MediaRecorder() on the MediaRecorder To stop recording the audio perform the following steps: Call the () method version on the MediaRecorderCall stop () method on the MediaRecorder Below is an excerpt from the MainActivity.java class that shows how to start and stop recording the audio How to play An audio recording in Android offers a MediaPlayer class that supports the ability for you to play audio and video files in your app. For a list of all formats supported by the MediaPlayer check out the supported support formats documentation. I've improved the app example created above for programmatic audio recording in Android to also include the ability to play your audio recording once you've stopped recording. It also allows you to stop playing audio while it is being played. In order to play an audio recording using MediaPlayer, you need to follow these simple steps. Create a new instance of the MediaPlayerSet the data source of the to the file you want to read using the setDataSource (String) method Use the setOnCompletionListener method (...) if you want to perform certain features once the audio file has completed playbackCall the preparation method () on the playerCall the start() method on the player below is an excerpt from the activity_main.xml layout file that has got some Buttons to use for playback of audio recording. Here is an excerpt from the MainActivity class.java which shows how to read and stop playing an audio recording. In addition to the sample code shared in this blog post, a public repository for this project has been created called AudioRecorder containing all the code covered above. Facebook Twitter Email WhatsApp link to how to use IBM Watson Speech to Text Service in an Android App link to how to use IBM Watson Text to Speech Service in an Android App report this ad