MediaRecorder.AudioSource | API reference | Android Developers

Archived: 2026-04-05 18:53:24 UTC

```
public final class MediaRecorder.AudioSource
extends Object
```

Defines the audio source. An audio source defines both a default physical source of audio signal, and a recording configuration. These constants are for instance used in `MediaRecorder.setAudioSource(int)` or `AudioRecord.Builder.setAudioSource(int)` .

Summary

Constants	
`int`	`CAMCORDER` Microphone audio source tuned for video recording, with the same orientation as the camera if available.
`int`	`DEFAULT` Default audio source *
`int`	`MIC` Microphone audio source
`int`	`REMOTE_SUBMIX` Audio source for a submix of audio streams to be presented remotely.
`int`	`UNPROCESSED` Microphone audio source tuned for unprocessed (raw) sound if available, behaves like `DEFAULT` otherwise.

int	VOICE_CALL Voice call uplink + downlink audio source Capturing from VOICE_CALL source requires the Manifest.permission.CAPTURE_AUDIO_OUTPUT permission.
int	VOICE_COMMUNICATION Microphone audio source tuned for voice communications such as VoIP.
int	VOICE_DOWNLINK Voice call downlink (Rx) audio source.
int	VOICE_PERFORMANCE Source for capturing audio meant to be processed in real time and played back for live performance (e.g karaoke).
int	VOICE_RECOGNITION Microphone audio source tuned for voice recognition.
int	VOICE_UPLINK Voice call uplink (Tx) audio source.

Inherited methods

From class java.lang.Object

Object	clone() Creates and returns a copy of this object.
boolean	equals(Object obj) Indicates whether some other object is "equal to" this one.

`void`	`finalize()` Called by the garbage collector on an object when garbage collection determines that there are no more references to the object.
`final Class<?>`	`getClass()` Returns the runtime class of this `Object`.
`int`	`hashCode()` Returns a hash code value for the object.
`final void`	`notify()` Wakes up a single thread that is waiting on this object's monitor.
`final void`	`notifyAll()` Wakes up all threads that are waiting on this object's monitor.
`String`	`toString()` Returns a string representation of the object.
`final void`	`wait(long timeoutMillis, int nanos)` Causes the current thread to wait until it is awakened, typically by being *notified* or *interrupted*, or until a certain amount of real time has elapsed.
`final void`	`wait(long timeoutMillis)` Causes the current thread to wait until it is awakened, typically by being *notified* or *interrupted*, or until a certain amount of real time has elapsed.
`final void`	`wait()` Causes the current thread to wait until it is awakened, typically by being *notified* or *interrupted*.

Constants

CAMCORDER

```
public static final int CAMCORDER
```

Microphone audio source tuned for video recording, with the same orientation as the camera if available.

Constant Value: 5 (0x00000005)

DEFAULT

```
public static final int DEFAULT
```

Default audio source *

Constant Value: 0 (0x00000000)

MIC

```
public static final int MIC
```

Microphone audio source

Constant Value: 1 (0x00000001)

REMOTE_SUBMIX

```
public static final int REMOTE_SUBMIX
```

Audio source for a submix of audio streams to be presented remotely.

An application can use this audio source to capture a mix of audio streams that should be transmitted to a remote receiver such as a Wifi display. While recording is active, these audio streams are redirected to the remote submix instead of being played on the device speaker or headset.

Certain streams are excluded from the remote submix, including `AudioManager.STREAM_RING`, `AudioManager.STREAM_ALARM`, and `AudioManager.STREAM_NOTIFICATION`. These streams will continue to be presented locally as usual.

Capturing the remote submix audio requires the `Manifest.permission.CAPTURE_AUDIO_OUTPUT` permission. This permission is reserved for use by system components and is not available to third-party applications.

.
Requires `Manifest.permission.CAPTURE_AUDIO_OUTPUT`

Constant Value: 8 (0x00000008)

UNPROCESSED

```
public static final int UNPROCESSED
```

Microphone audio source tuned for unprocessed (raw) sound if available, behaves like `DEFAULT` otherwise.

Constant Value: 9 (0x00000009)

VOICE_CALL

```
public static final int VOICE_CALL
```

Voice call uplink + downlink audio source

Capturing from `VOICE_CALL` source requires the `Manifest.permission.CAPTURE_AUDIO_OUTPUT` permission. This permission is reserved for use by system components and is not available to third-party applications.

Constant Value: 4 (0x00000004)

VOICE_COMMUNICATION

```
public static final int VOICE_COMMUNICATION
```

Microphone audio source tuned for voice communications such as VoIP. It will for instance take advantage of echo cancellation or automatic gain control if available.

Constant Value: 7 (0x00000007)

VOICE_DOWNLINK

```
public static final int VOICE_DOWNLINK
```

Voice call downlink (Rx) audio source.

Capturing from `VOICE_DOWNLINK` source requires the `Manifest.permission.CAPTURE_AUDIO_OUTPUT` permission. This permission is reserved for use by system components and is not available to third-party applications.

Constant Value: 3 (0x00000003)

VOICE_PERFORMANCE

```
public static final int VOICE_PERFORMANCE
```

Source for capturing audio meant to be processed in real time and played back for live performance (e.g karaoke).

The capture path will minimize latency and coupling with playback path.

Constant Value: 10 (0x0000000a)

VOICE_RECOGNITION

```
public static final int VOICE_RECOGNITION
```

Microphone audio source tuned for voice recognition.

Constant Value: 6 (0x00000006)

VOICE_UPLINK

```
public static final int VOICE_UPLINK
```

Voice call uplink (Tx) audio source.

Capturing from `VOICE_UPLINK` source requires the `Manifest.permission.CAPTURE_AUDIO_OUTPUT` permission. This permission is reserved for use by system components and is not available to third-party applications.

Constant Value: 2 (0x00000002)
