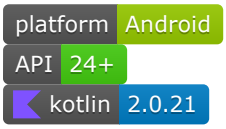
SpeechMike Ambient



The SpeechMike Ambient is engineered specifically to capture high-fidelity, single and multi-speaker audio in real-world environments - perfect for AI transcription, conversational AI and ambient scribe scenarios in clinical, and virtual assistant functions. Its superior input quality significantly enhances AI model accuracy, making it ideal for generating clinical notes, transcriptions, and real-time documentation More information is available here.

Bluetooth SDK library

The SpeechMike Ambient Android SDK provides a comprehensive solution for integrating Philips SpeechMike Ambient devices with Android applications via Bluetooth Low Energy (BLE). This SDK enables real-time audio streaming, device management, and advanced audio processing capabilities for professional dictation and voice recording applications. The SDK itself is distributed as a prebuilt AAR library (written in Kotlin).

🔒 Note:
The SDK source code is maintained in a private repository.
This public repository is intended to showcase integration, usage, and behavior of the SDK through a demo application.

📦 Project Structure

```
root
├── app/                                # Demo Android application
│   ├── src/
│   ├── libs/sma-lib-<version>.aar      # Library file
│   └── build.gradle
├── build.gradle
└── README.md
```

🧩 SDK Overview

- **Language:** Kotlin

- **Distribution:** `.aar` Android library
- **Communication:** Bluetooth Low Energy (BLE)
- **Target Device:** Philips SpeechMike Ambient

The SDK abstracts:

- BLE scanning and connection
- GATT services and characteristics
- Device-specific commands and events

This allows client applications to focus only on **business logic**, not low-level BLE handling.

📥 SDK Integration (AAR)

The SDK is included as a local `.aar` dependency. Follow these steps to include it in your project:

1️⃣ Add the AAR File

Place the `.aar` file inside a local module or `libs/` directory.

Example structure:

```
libs/sma-lib-<version>.aar
```

2️⃣ Gradle Configuration

app/build.gradle

```
implementation(files("libs/sma-lib-<version>.aar"))
```

🔐 Permissions & Requirements

Component	Version
Min SDK	API 24 (Android 7.0)
Target SDK	API 36 (Android 15)
Compile SDK	API 36
Kotlin	2.0.21+
Java	11
Gradle	8.13.0+

> ⚠️ BLE and runtime permission handling is implemented in the demo app.

📱 Demo App

The repository includes a fully functional demo application showcasing all SDK features.
You can try the demo app by downloading the APK from the releases section of this repository.

Features Demonstrated

- ✅ BLE device scanning with filtering
- ✅ Device connection and disconnection
- ✅ Real-time audio recording
- ✅ Audio mode switching
- ✅ LED control
- ✅ Power event monitoring
- ✅ Out of reach event monitoring
- ✅ PCM to WAV conversion
- ✅ File management and playback

Running the Demo

1. Clone the repository
2. Open in Android Studio
3. Build and run the `app` module
4. Grant required permissions
5. Scan for your SpeechMike Ambient device
6. Connect and start recording

Demo App Structure

```
app/
├── src/main/java/com/philips/sma/
│   ├── SmaApplication.kt              # Application class
│   ├── ui/
│   │   ├── MainActivity.kt            # Device scanning and connection
│   │   ├── RecorderActivity.kt        # Audio recording interface
│   │   └── DevicesAdapter.kt          # Device list adapter
│   └── util/
```

```
|       ├── Constants.kt              # App constants
|       └── Utils.kt                  # Helper utilities
```

📚 Documentation

Core Classes

`SpeechMikeAmbientSdk`

Main SDK class - Singleton pattern for managing device operations.

```
// Get instance
val sdk = SpeechMikeAmbientSdk(context)

// Main methods
sdk.scanForDevices(nameFilter: String?, pairingOnly: Boolean, onDeviceFou
sdk.stopScan(onStopped: (() -> Unit)?)
sdk.connectToDevice(device: BluetoothDevice, onConnectionResult: (Connect
sdk.disconnectDevice(onDisconnected: ((Boolean) -> Unit)?)
sdk.setListener(listener: SpeechMikeAmbientListener)
sdk.startAudioStreaming()
sdk.stopAudioStreaming()
sdk.setAudioMode(mode: AudioMode)
sdk.getAudioMode(): AudioMode
sdk.setLed(greenState: LedState, redState: LedState)
```

Additional helper methods

```
sdk.getFirmwareVersion(): String //Retrieves the firmware version of the
sdk.isConnected(): Boolean //Check if a device is currently connected.
sdk.resetConnectionForBondRecovery(onComplete: (() -> Unit)?) //Resets th
sdk.restartBluetoothAdapter(onComplete: (() -> Unit)?): Boolean //Restart
sdk.setClearCacheOnDisconnect(clearCache: Boolean) // Configures whether
sdk.release() //Releases all resources held by the SDK.
```

`SpeechMikeAmbientListener`

Interface for receiving SDK events.

```
interface SpeechMikeAmbientListener {
    fun onAudioFrame(audioStream: AudioStream)
    fun onRecordingStarted()
    fun onRecordingStopped()
    fun onButtonEvent(buttons: MutableList<ButtonEvent>)
    fun onAudioModeChanged(mode: AudioMode)
    fun onDevicePowerEvent(event: PowerEvent)
    fun onDeviceDisconnected()
    fun onDeviceOutOfReach()
}
```

📄 Limitations

- SDK source code is not included in this repository
- Demo app focuses on core BLE functionality only
- UI is intentionally minimal

🐛 Troubleshooting

Common Issues

Device Not Found During Scan

- Ensure Bluetooth is enabled
- Check location permissions (Android < 12)
- Verify BLUETOOTH_SCAN permission (Android 12+)
- Make sure the device is powered on and not connected to another device

Connection Fails

- Check BLUETOOTH_CONNECT permission
- Ensure device is within range (typically 10 meters)
- Try resetting the SpeechMike Ambient device
- Verify the device is not in deep sleep mode
- Follow the guidelines shown in the application

No Audio Frames Received

- Verify `startAudioStreaming()` was called after connection
- Check that the recording button on the device was pressed
- Ensure the listener is set before starting streaming

- Verify the device has sufficient battery

Audio Quality Issues

- Experiment with different `AudioMode` settings
- Check sample rate configuration
- Ensure proper PCM to WAV conversion parameters
- Monitor connection quality

📊 Technical Specifications

BLE Characteristics

Service	Characteristic	Type	Description
Audio Service	Audio Stream	Notify	Real-time audio data
Control Service	Audio Mode	Read/Write	Audio mode configuration
Control Service	LED Control	Write	LED state control
Status Service	Power Events	Notify	Device power state changes
Status Service	Out Of Reach Events	Notify	Device connectivity state changes

Audio Specifications

Parameter	Value
Sample Rates	8 kHz, 16 kHz
Bit Depth	16-bit
Frame Size	56 bytes (mono), 112 bytes (stereo)

🏆 Acknowledgments

- Designed for [Philips SpeechMike Ambient] devices
