



CogInsight

Chatbot SDK for Android

COPYRIGHT MindwareWorks.

121-839 서울특별시 월드컵북로 12길 19 대성빌딩 2층

전화 (070) 4046-6752 • 팩스 (02) 3406-6752

담당자 문서 제·개정 이력

버전	날짜	내용	담당자
3.0.1	2022-08-22	최초 작성	강신준

목차

목차	2
개요	4
요구 사항	5
지원 환경	5
인증 정보	5
풀더 구조	5
Chatbot SDK	6
1. 프로젝트 설정	6
1.1. build.gradle에 직접 추가하는 방법	6
2. 초기화	7
2.1. ChatService 생성	7
2.1.1. 챗봇 UI 없이 생성하는 경우	7
2.1.2. 챗봇 UI를 사용하여 생성하는 경우	7
2.2. ChatService 실행	7
2.2.1. ChatSetting	7
2.2.1.1. ChatSetting 생성	8
2.2.1.2. Version 관리	8
2.2.1.3. 초기 입력 데이터 설정	8
2.2.1.4. 인증 데이터 설정	9
2.2.2. ChatViewEventListener	9
2.2.2.1. onError(ChatError error)	9
2.2.2.2. onChatCreate	10
3. STT (Speech to Text) 처리 방법	11
3.1. STT 서비스 시작	11
3.2. 음성 데이터 전달	11
3.2.1. byte[]로 전달	11
3.2.2. Base64 인코딩 텍스트로 전달	11
3.3. STT 서비스 종료	11
3.4. 이벤트 핸들러 (ChatSpeechEventListener)	12
3.4.1. onStart	12
3.4.2. onStop	12
3.4.3. onTranscript	12
3.4.4. onVoice	13
4. Dialog Message	14
4.1. 메시지 전송	14

4.1.1. ChatSpeechText를 사용	14
4.1.2. ChatData를 사용	14
4.1.2.1. setText	14
4.1.2.2. setReset	14
4.1.2.3. addData	15
4.2. 이벤트 핸들러	15
4.2.1. onReceive	15
5. TTS (Text to Speech)	16
5.1. TTS 요청	16
5.1.1. ChatDialogMessage 사용	16
5.1.2. 직접 텍스트를 입력	16
5.1.3. 이벤트 핸들러	16
6. 디버그 로깅	17
7. 에러 메시지	18
7.1. 인증 처리 중 발생하는 에러	18
7.2. Speech 서비스 동작 중 발생하는 에러	18
7.3. Dialog 서비스 동작 중 발생하는 에러	19
7.4. 기타	19
Reference	20
Appendix	21



개요

본 문서는 Mindwareworks 의 CogInsight 솔루션의 Chatbot SDK 사용 방법을 설명하고 있습니다.

본 문서는 기술적인 오류나 구문의 오류를 포함하고 있을 수 있습니다. 당사는 본 문서에 포함된 정보의 정확성을 유지하기 위해 최선을 다할 것이나, 본 문서의 기술적 오류, 잘못된 정보가 포함되어 있지 않다는 것을 보증하지 않습니다.

본 문서는 특별한 언급 없이 지속적으로 수정과 보완할 것이나 본 문서에 기술된 정보로 인하여 발생할 수 있는 직접적 혹은 간접적 손해, 데이터, 프로그램 기타 무형의 재산에 관한 손실, 사용 이익의 손실 등에 관하여 비록 이와 같은 손해 가능성에 대해 사전에 알고 있었다고 해도 손해 배상 등 기타 책임을 지지 않습니다.

사용자는 본 문서를 구입하거나, 전자문서로 다운로드 받거나, 사용을 시작함으로써, 본 사항에 명시된 내용을 이해하며, 이에 동의하는 것으로 간주합니다. 또한 본 내용이 이전의 문구나 기타 고지에 우선 하는 것임을 인정합니다.

요구 사항

CogInsight 솔루션의 **Console** 사이트에서 SDK 사용에 필요한 정보를 확인할 수 있습니다.

지원 환경

1. 지원 OS : Android(minSDKVersion:18)
2. 개발환경 : Android Studio

인증 정보

CogInsight 솔루션 > **Console** 접속 > Domain > 인증키 관리 메뉴에서 확인

name	type	required	description
projectId	string	Y	챗봇 ID
apiKey	string	Y	Domain API Key

풀더 구조

name	description
chatbot-android-v3.0.1-sdk.aar	SDK 라이브러리
chatbot-android-v3.0.1-doc.zip	Javadoc
chatbot-android-v3.0.1-demo.zip	TestClient

※ TestClient 는 CogInsight 개발 서버 (v2.aiware.io) 의 테스트 챗봇에 연결되어 동작합니다.
네트워크 환경에 따라, 방화벽 오픈 (포트:443) 이 필요할 수 있습니다.



Chatbot SDK

1. 프로젝트 설정

chatbot-android-vX.X.X-release.aar 파일을 프로젝트에 추가합니다.

1.1. build.gradle에 직접 추가하는 방법

App 의 build.gradle 파일에 아래와 같이 .aar 파일을 추가합니다.

```
Implementation files("libs/chatbot-android-vX.X.X-release.aar")
```

2. 초기화

2.1. ChatService 생성

SDK에서 제공하는 모든 기능은 ChatService를 생성한 이후에 활성화됩니다. ChatService를 생성하는 방법은 다음과 같습니다.

2.1.1. 챗봇 UI 없이 생성하는 경우

챗봇 UI가 필요 없는 경우 ChatService 객체를 생성시에 Activity의 Context를 인자로 전달해줍니다.

```
ChatService chatService = new ChatService(context);
```

2.1.2. 챗봇 UI를 사용하여 생성하는 경우

Activity Layout에 ChatView를 사용해서 화면에 챗봇을 노출하여 사용하는 경우, ChatView를 인자로 전달해줍니다.

```
ChatView chatView = findViewById(R.id.chat_view);
ChatService chatService = new ChatService(chatView);
```

2.2. ChatService 실행

ChatService 생성 이후에 start() 메서드를 호출해서 서비스를 동작 시킵니다.

```
ChatSetting setting = ...;
ChatViewEventListener listener = ...;

chatService.start(setting, listener);
```

ChatSetting과 ChatViewEventListener에 대한 설명은 다음과 같습니다.

2.2.1. ChatSetting

ChatSetting은 ChatService 동작 시 필요한 데이터를 설정하기 위해 사용합니다.

2.2.1.1. ChatSetting 생성

ChatSetting은 ChatSetting.Builder를 통해 생성합니다. Service URL과 Project ID는 필수 항목이므로 가장 기본적인 생성 방법은 아래와 같습니다.

```
ChatSetting setting = new ChatSetting.Builder()  
    .setServerUrl(serverUrl)  
    .setProjectId(projectId).build();
```

2.2.1.2. Version 관리

챗봇은 배포방식으로 동작합니다. 즉, 특정 버전을 지정해서 동작하게 되며, 버전을 지정하지 않으면 최근에 배포한 버전으로 동작합니다. 이로 인해 개발 중인 챗봇을 확인할 때는 따로 개발용 버전이라는 것을 설정해주어야 현재 개발중인 챗봇 동작을 테스트 할 수 있습니다. 개발 버전은 다음과 같이 버전을 “dev”로 설정합니다.

```
ChatSetting setting = new ChatSetting.Builder()  
    ...  
    .setVersion(“dev”)  
    .build();
```

상용 배포 시에는 반드시 버전 설정을 없앤 후에 배포해야 합니다.

2.2.1.3. 초기 입력 데이터 설정

챗봇 시작 시 초기 입력 데이터를 설정할 수 있습니다. 다음과 같이 ChatData 를 생성해서 ChatSetting에 추가합니다.

```
ChatData inputData = ChatData.create()  
inputData.setText(“Input Text”);  
inputData.addData(“data-1”, “value-1”);  
  
ChatSetting setting = new ChatSetting.Builder()  
    ...  
    .setInputData(inputData)
```

```
.build();
```

2.2.1.4. 인증 데이터 설정

챗봇은 기본적으로 누구나 접속할 수 있는 서비스입니다. 만약 특정 사용자를 지정해야 한다면, 인증 설정을 해줘야 합니다. 인증 데이터를 설정하려면, 반드시 User Key와 Domain API Key를 입력해 주어야 합니다.

```
ChatSetting.AuthParams authParams =  
    ChatSetting.AuthParams.create(userKey, apiKey)  
        .addSecureItem(“secure-1”, “secure-value-1”)  
  
ChatSetting setting = new ChatSetting.Builder()  
    ...  
    .setAuthParams(authParams)  
    .build();
```

2.2.2. ChatViewEventListener

ChatService 동작 중 발생하는 여러 이벤트를 감지하여 처리하기 위해 사용합니다. 가장 빈번하게 사용하는 주요 메서드는 onError와 onChatCreate입니다.

2.2.2.1. onError(ChatError error)

챗봇 동작 중 발생하는 모든 문제는 onError 메서드를 통해 전달됩니다. 인자로 전달되는 error의 내용을 통해 어떤 문제가 발생했는지 확인할 수 있습니다.

- **Error Code:** error.getCode() 메서드를 통해 문제에 대한 코드 값을 확인할 수 있습니다.
- **Request URL:** error.getRequestUrl() 메서드를 통해 문제에 대한 URL을 확인할 수 있습니다. URL은 실제 웹 페이지에 대한 것일 수도 있고, 내부 동작 중 어떤 과정인지를 나타낼 수도 있습니다. 예를 들어 인증 처리 과정 중 인증 토큰에 문제가 발생하면 Request URL은 auth/token이 됩니다.

- **Error Description:** Error.getDescription() 메서드를 통해 확인할 수 있으며, 문제에 대한 간단한 메시지입니다. 보통 문제 발생 시 사람이 바로 확인하는 용도로 사용합니다.

2.2.2.2. onChatCreate

ChatService 시작 후 챗봇 초기화가 완료되면 호출되는 메서드입니다.

3. STT (Speech to Text) 처리 방법

ChatService가 제공하는 Speech 서비스를 통해 STT 처리를 할 수 있습니다. STT는 실시간으로 음성 데이터를 전달받아 이를 텍스트로 변환하여 이벤트 핸들러를 통해 전달합니다.

3.1. STT 서비스 시작

Speech 서비스를 아래와 같이 시작합니다.

```
chatService.getSpeech().startListen();
```

초기화가 완료되어 음성 데이터를 받을 준비가 되면 이벤트 핸들러의 onStart() 메서드가 실행됩니다.

3.2. 음성 데이터 전달

텍스트로 변환을 원하는 음성 데이터를 다음과 같이 전달합니다.

3.2.1. byte[]로 전달

실시간으로 수집되거나 기타 다른 경로로 확보된 음성 데이터를 byte[] 형식으로 전달합니다.

```
byte[] bytes = ...;
chatService.getSpeech().sendBytes(bytes);
```

3.2.2. Base64 인코딩 텍스트로 전달

실시간으로 수집되거나 기타 다른 경로로 확보된 음성 데이터를 Base64 텍스트로 전달합니다.

```
String base64 = ...;
chatService.getSpeech().sendBase64(base64);
```

3.3. STT 서비스 종료

STT 서비스는 기본적으로 하나의 문장이 완성되면 자동으로 종료됩니다. 이는 isFinal 값으로 확인할 수 있습니다. isFinal은 이벤트 핸들러 항목에서 자세히 설명합니다.

이외에 아래와 같이 동작중인 서비스를 종료 시킬 수 있습니다.

```
chatService.getSpeech().stopListen();
```

종료 처리가 완료되면, 이벤트 핸들러의 `onStop()` 메서드가 호출됩니다.

3.4. 이벤트 핸들러 (ChatSpeechEventListener)

Speech 서비스의 결과물을 전달받기 위해서는 이벤트 핸들러를 등록해야 합니다.

```
chatService.getSpeech().setSpeechEventListener(  
    new ChatSpeechEventListener() {  
        @Override  
        public void onTranscript(ChatSpeechText text) { ... }  
  
        @Override  
        public void onStart() { ... }  
  
        @Override  
        public void onStop() { ... }  
  
        @Override  
        public void onVoice(byte[] data) { ... }  
    }  
);
```

3.4.1. onStart

Speech 서비스가 동작할 준비가 완료되면 호출됩니다. 이때부터 Speech 서비스가 제공하는 기능을 사용할 수 있습니다.

3.4.2. onStop

Speech 서비스가 종료되면 호출됩니다. 관련 리소스를 정리할 수 있습니다.

3.4.3. onTranscript

음성 데이터로부터 변환된 텍스트를 전달하기 위해 호출됩니다. Speech 서비스의 STT는 실시간으로 음성 데이터를 변환합니다. 인자로 전달되는 `ChatSpeechText`의 데이터는 다음과 같습니다.

- `text.getTranscript()`

현재까지 변환된 텍스트입니다.

- **text.getSpeechId()**

현재 동작중인 STT 세션의 아이디입니다.

- **text.isFinal()**

하나의 완전한 문장이 완성되었는지 여부입니다. STT 세션은 isFinal이 true가 될 때까지 음성 데이터를 입력할 수 있습니다. 최종적으로 하나의 문장이 완성되면 isFinal은 true를 반환하고 STT 세션은 종료됩니다. 이때에 ChatSpeechEventListener의 onStop()을 호출됩니다.

ChatSpeechText는 이후 Dialog의 입력 메시지로 사용될 수 있습니다.

3.4.4. onVoice

TTS의 결과로 음성 데이터를 전달하기 위해 호출됩니다. 이와 관련해서는 TTS 항목에서 자세히 설명합니다.

4. Dialog Message

Dialog는 챗봇과 대화를 처리하는 방식을 정의한 서비스입니다. 즉, 챗봇에게 메시지를 전송하고 그 결과를 받아오는 과정을 처리합니다.

4.1. 메시지 전송

챗봇에게 메시지를 전송하는 방식은 직접 입력 텍스트 및 데이터를 작성하거나 STT의 결과인 ChatSpeechText를 입력으로 사용하는 것입니다.

4.1.1. ChatSpeechText를 사용

STT 처리 후 ChatSpeechEventHandler의 onTranscript 메서드가 호출되면서 전달되는 ChatSpeechText 데이터를 다음과 같이 직접 입력 메시지로 전달합니다.

```
chatService.getDialog().inputMessage(chatSpeechText);
```

4.1.2. ChatData를 사용

직접 입력 메시지를 만들어 사용하려면 ChatData를 생성해서 전달합니다. 사용법은 다음과 같습니다.

```
ChatData data = ChatData.create()
    .setText(text)
    .setReset(true)
    .addData(“textData”, “Text”)
    .addData(“numberData”, 100)
    .addData(“boolData”, true);

chatService.getDialog().inputMessage(data);
```

4.1.2.1. setText

입력 문구로 사용할 텍스트를 설정합니다.

4.1.2.2. setReset

현재 챗봇과의 대화 흐름을 초기화합니다. 특별한 조건이 없다면 “Welcome” 메시지가 결과로 내려옵니다.

4.1.2.3. addData

입력 메시지에 특정 형식의 데이터를 함께 보낼 때 사용합니다. 챗봇 시나리오에 따라 다양한 형식의 데이터를 보낼 수 있습니다.

4.2. 이벤트 핸들러

메시지 전송 후 챗봇의 응답을 받기 위해 이벤트 핸들러를 설정합니다.

```
chatService.getDialog().setDialogEventListener(  
    new ChatDialogEventListener() {  
        @Override  
        public void onReceive(ChatDialogMessage message) { ... }  
    }  
);
```

4.2.1. onReceive

챗봇이 응답 메시지를 보내면 호출됩니다. 인자로 전달되는 ChatDialogMessage의 데이터는 다음과 같습니다.

- **message.getMessageId()**
응답 메시지의 아이디입니다.
- **message.getOutput()**
JSONObject 타입의 응답 메시지입니다.
- **message.getAt()**
응답 메시지의 시간입니다.

ChatDialogMessage 데이터는 이후 TTS의 입력으로 사용될 수 있습니다.

5. TTS (Text to Speech)

TTS는 STT와 마찬가지로 Speech 서비스를 통해 처리합니다.

5.1. TTS 요청

TTS 요청 시 ChatDialogMessage 데이터를 사용하거나 직접 텍스트를 입력할 수 있습니다.

5.1.1. ChatDialogMessage 사용

Dialog 메시지 입력 결과인 ChatDialogMessage를 TTS의 입력으로 사용합니다.

```
chatService.getSpeech().sendText(chatDialogMessage);
```

5.1.2. 직접 텍스트를 입력

TTS 변환이 필요한 텍스트를 직접 입력으로 사용합니다.

```
chatService.getSpeech().sendText("텍스트");
```

5.1.3. 이벤트 핸들러

위 3.4) 항목에서 설정한 Speech의 서비스의 이벤트 핸들러를 통해 TTS의 결과가 반환됩니다.

```
@Override  
public void onVoice(byte[] data) { ... }
```

입력한 데이터에 대한 전체 음성 데이터를 byte[] 형태로 전달합니다.

6. 디버그 로깅

챗봇 서비스 동작 중 로그 메시지를 확인하려면, 아래와 같이 로그 메시지를 활성화시켜줍니다.

```
ChatLog.setDebug(true);
```

7. 에러 메시지

챗봇 서비스 동작 중 에러가 발생하면, start() 호출 시에 전달한 ChatViewEventListener의 onError() 메서드가 호출됩니다.

상황에 따른 에러 내용은 다음과 같습니다.

7.1. 인증 처리 중 발생하는 에러

Code	URL	Description
451	auth/public-key	인증 처리를 위해 암호화를 수행할 때 잘못된 PUBLIC_KEY로 인해 발생한 에러입니다.
452	auth/encrypt	데이터 암호화 처리 중 발생한 에러입니다.
461	auth/token	데이터 암호화를 위해 사용된 토큰에 문제가 있을 때 발생하는 에러입니다.
462	auth/token	데이터 암호화를 위해 사용된 토큰이 비어 있을 때 발생하는 에러입니다.
472	auth/request-body	인증 요청을 위한 HTTPS 처리 중 JSON 파싱에 문제가 있을 때 발생하는 에러입니다.

7.2. Speech 서비스 동작 중 발생하는 에러

Code	URL	Description
480	speech	Speech 서비스 동작 중 발생하는 일반적인 에러입니다.
481	speech/parse-json	JSON 처리 중 발생하는 에러입니다.
482	speech/already-started	Speech 서비스가 이미 시작 중인데 다시 시작을 하려고 할 때 발생하는 에러입니다.
483	speech/socket	음성 데이터 전달을 위한 소켓 통신 중 발생하는 에러입니다.
484	speech/cannot-start	Speech 서비스를 시작할 수 없을 때 발생하는 에러입니다.

485	speech/already-sent-message	Dialog 입력 메시지로 이미 사용한 ChatSpeechText를 다시 입력 메시지로 사용한 경우에 발생합니다.
-----	-----------------------------	---

7.3. Dialog 서비스 동작 중 발생하는 에러

Code	URL	Description
490	dialog	Dialog 서비스 동작 중 발생하는 일반적인 에러입니다.
491	dialog/parse-json	JSON 처리 중 발생하는 에러입니다.
492	dialog/input-message	입력 메시지를 처리하는 과정에서 발생하는 에러입니다.

7.4. 기타

챗봇 서비스는 WebView를 기반으로 동작하고 있습니다. 이에 여러 네트워크 환경이나 서버 오류로 인해 발생하는 에러들이 onError를 통해 전달됩니다. 이때에는 해당 에러가 발생한 URL과 함께 전달되는 Description 내용을 확인하여 개별적으로 처리해야 합니다.

Reference

별첨

Appendix