

# 6주차(24.04.11)

- [https://docs.aws.amazon.com/ko\\_kr/sdk-for-javascript/v3/developer-guide/lex-bot-example.html](https://docs.aws.amazon.com/ko_kr/sdk-for-javascript/v3/developer-guide/lex-bot-example.html)
- <https://github.com/aws-samples/aws-workshop/HelloWorldBot.md>

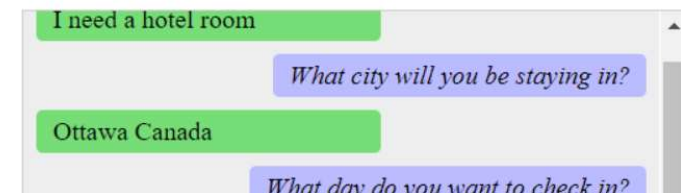
## Amazon Lex 챗봇 빌드

[PDF](#) | [RSS](#)

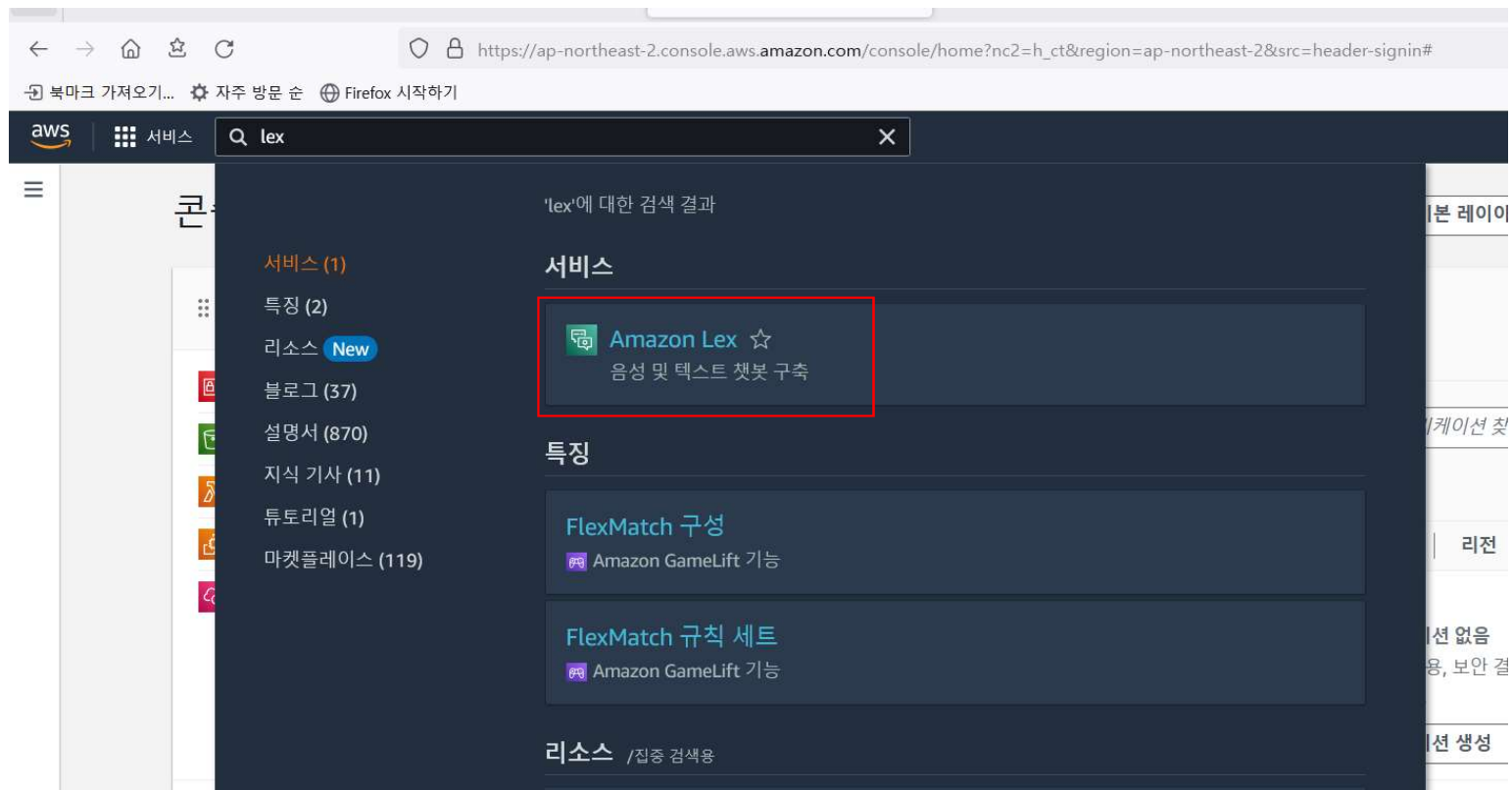
웹 애플리케이션 내에 Amazon Lex 챗봇을 생성하여 웹 사이트 방문자의 참여를 유도할 수 있는 Lex 챗봇은 사람과 직접 접촉하지 않고도 사용자와 온라인 채팅 대화를 수행하는 기능입니다. 림은 호텔 객실 예약과 관련하여 사용자의 참여를 유도하는 Amazon Lex 챗봇을 보여줍니다.

## Amazon Lex - BookTrip

This multiple language chatbot shows you how easy it is to incorporate [Amazon Lex](#) into your web apps. Try it out.



# Amazon Lex



# 봇 생성

The screenshot shows the Amazon Lex console interface. The top navigation bar includes the AWS logo, a search bar, and various utility icons. The left sidebar displays the 'Amazon Lex' title and a navigation menu with options like '봇', 'Bot templates', 'Networks of bots', 'Test workbench', and '관련 리소스'. The main content area is titled 'Lex > 봇' and features a '봇 (0)' section with an 'Info' link. This section includes a search bar, a table with columns for '이름', '설명', '상태', '최신 버전', and 'Last updated', and a message '봇을 찾을 수 없음' with a '봇 생성' button. Below this is a '가져오기/내보내기 기록 (0)' section with an 'Info' link, a search bar, a table with columns for '유형', 'Bot', '상태', '오류', '마지막 업데이트 날짜', '파일', and '버전', and a message '가져오기/내보내기 레코드를 찾을 수 없음'. A red box highlights the '봇 생성' button in the top right corner of the '봇 (0)' section.

aws 서비스 검색 [알트+S]

Amazon Lex X

봇

Bot templates New

Networks of bots New

▼ Test workbench New

Test sets

Test results

▶ 관련 리소스

Lex > 봇

봇 (0) Info

작업 ▼ **봇 생성**

Search 봇

< 1 > ⚙

이름 ▲	설명 ▼	상태 ▼	최신 버전 ▼	Last updated ▼
봇을 찾을 수 없음				

봇 생성

가져오기/내보내기 기록 (0) Info

선택 항목 삭제

Search 가져오기/내보내기 기록

< 1 > ⚙

유형 ▼	Bot ▼	상태 ▼	오류 ▼	마지막 업데이트 날짜 ▲	파일	버전
가져오기/내보내기 레코드를 찾을 수 없음						

# booktrip예제 생성

언어 선택

한국어(KR) ▼

설명 - 선택 사항

최대 200자.

음성 상호 작용

봇이 사용자와 상호 작용하는 데 사용하는 텍스트 투 스피치 음성입니다.

Seoyeon ▼

음성 샘플

안녕하세요 제 이름은 Seoyeon 입니다. 무엇을 도와드릴까요?

재생

의도 분류 신뢰도 점수 임계값

0.40

최소: 0.00, 최대: 1.00.

취소

다른 언어 추가

완료

# Build

The screenshot shows the Amazon Lex console interface. At the top, there's a navigation bar with the AWS logo, '서비스' (Services), a search bar, and a '[알트+S]' shortcut. Below this, the 'Amazon Lex' header is visible. On the right side of the header, there are buttons for 'Draft version', '한국어(KR)', and '구축되지 않음' (Not built). A status message indicates '한국어(KR) has not built changes.' To the right of this message is the 'Build' button, which is highlighted with a red 'X' and a red circle. Next to it is a 'Test' button. Below the header, there's a section for '코드 후크 - 선택 사항' (Code hooks - optional) with a checkbox for '초기화 및 검증에 Lambda 함수 사용' (Use Lambda function for initialization and validation). The main area is divided into 'Editor' and 'Visual builder' tabs. On the right side, there's a '초안 버전 테스트' (Draft version test) window. This window shows a chat conversation with a user asking '체크인할 날짜는 언제입니까?' (What date will I check in?) and '얼마나 오랫동안 숙박할 예정입니까?' (How long will I stay?). The test results show '전체 테스트 준비 완료' (All test preparation complete) and '메시지 입력' (Message input).

Build

빌드가 끝나면 test로 실행해볼수 있음.

# Node.js install

```
C:\Windows\System32\cmd × + v
Microsoft Windows [Version 10.0.22631.3447]
(c) Microsoft Corporation. All rights reserved.

E:\05.VSCode\chatbot>npm init

{
  "name": "chatbot",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" &&"
  },
  "author": "",
  "license": "ISC"
}

Is this OK? (yes) y

E:\05.VSCode\chatbot>npm install aws-sdk --save
```


# Node.js install

## Step 2: Set Up the Node.js Project

Now, let's set up a Node.js project to integrate the chatbot into our website:

1. Create a new directory for your project and navigate into it.
2. Initialize a new Node.js project by running the following command in your terminal:

```
npm init -y
```

 Copy code

3. Install the AWS SDK for JavaScript using the following command:

```
npm install aws-sdk
```

 Copy code

# 별칭

Amazon Lex

✕

봇

Bot templates [New](#)

Networks of bots [New](#)

BookTrip

봇 버전

초안 버전

모든 언어

▼ 한국어(KR)

의도

슬롯 유형

▼ 배포

별칭

채널 통합

▼ 분석 [New](#)

Overview

봇 세부 정보

편집

이름


BookTrip

설명

처음만드는 Lex 챗 봇 입니다.

ID: 6QTTIHSOZX

언어 추가 [Info](#)



봇이 대화에 참여할 수 있도록 언어를 추가합니다. 봇의 각 언어는 목표 달성을 위한 의도, 대화 시작을 위한 샘플 표현 허용, 추가 정보 수집을 위한 슬롯 및 사용자 요청 완료를 위한 이행으로 구성되어야 합니다.

언어 보기

✔ 언어 1개

배포를 위한 버전 및 별칭 생성 [Info](#)



# Index.html

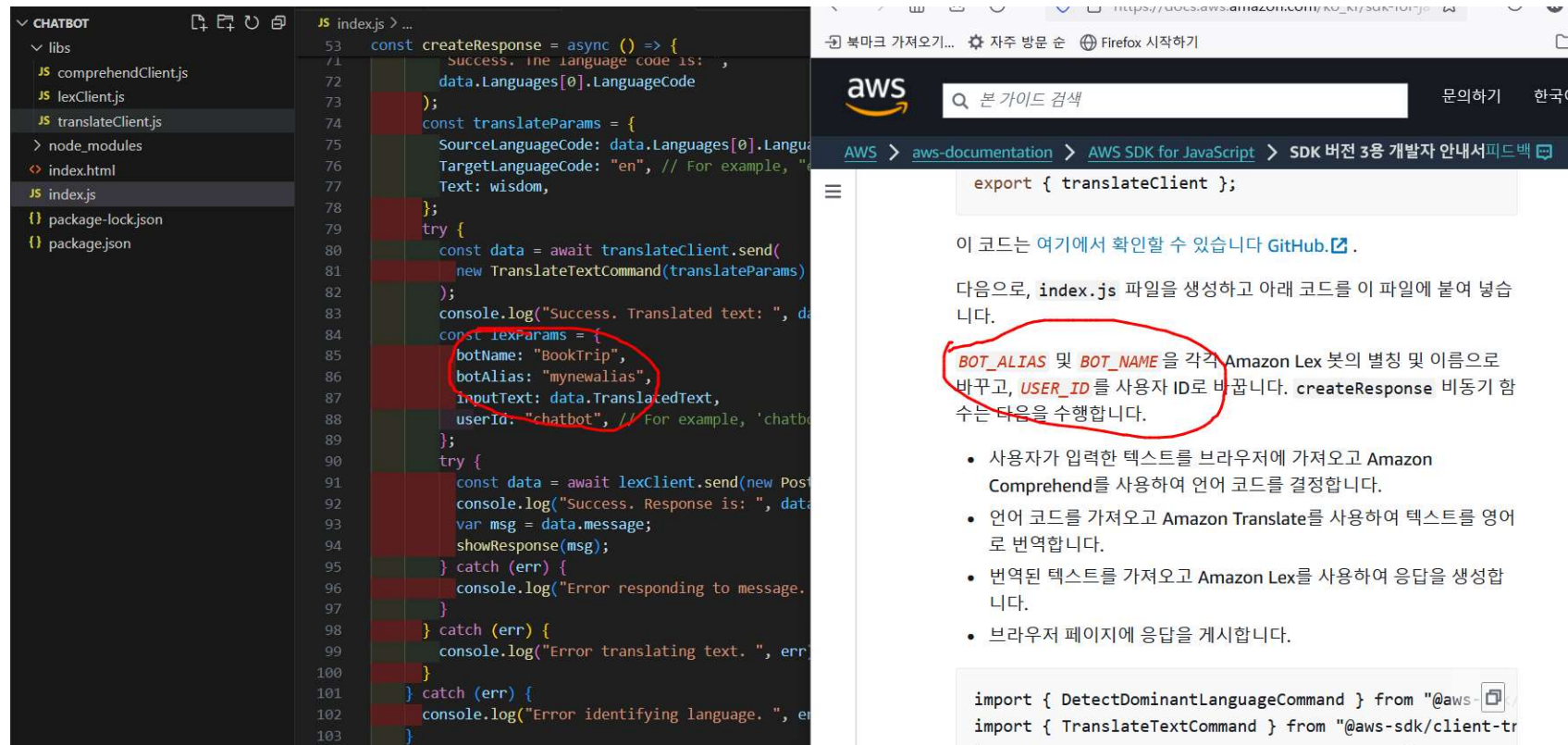
## HTML 생성

`index.html` 이라는 이름의 파일을 만듭니다. 아래 코드를 복사하여 `index.html` 에 붙여 넣습니다. 이 HTML은 `main.js` 를 참조합니다. 이는 필수 AWS SDK for JavaScript 모듈이 포함된 `index.js` 의 번들 버전입니다. HTML 생성에서 이 파일을 생성합니다. 또한 `index.html` 은 스타일을 추가하는 `style.css` 도 참조합니다.

```
<!doctype html>
<head>
  <title>Amazon Lex - Sample Application (BookTrip)</title>
  <link type="text/css" rel="stylesheet" href="style.css" />
</head>

<body>
  <h1 id="title">Amazon Lex - BookTrip</h1>
  <p id="intro">
    This multiple language chatbot shows you how easy it
  <a
    href="https://aws.amazon.com/lex/"
    title="Amazon Lex (product)"
    target="_new"
  >Amazon Lex</a>
  >
```

# BOT\_ALIAS, BOT\_NAME 변경



The image shows a code editor on the left and a web browser on the right. The code editor displays the `index.js` file with the following code:

```
53 const createResponse = async () => {
71   Success. the language code is: ",
72   data.Languages[0].LanguageCode
73 };
74 const translateParams = {
75   SourceLanguageCode: data.Languages[0].LanguageCode,
76   TargetLanguageCode: "en", // For example, "en"
77   Text: wisdom,
78 };
79 try {
80   const data = await translateClient.send(
81     new TranslateTextCommand(translateParams)
82   );
83   console.log("Success. Translated text: ", data.Text);
84   const lexParams = {
85     botName: "BookTrip",
86     botAlias: "mynewalias",
87     inputText: data.TranslatedText,
88     userId: "chatbot", // For example, 'chatbot'
89   };
90   try {
91     const data = await lexClient.send(new PostTextCommand(lexParams));
92     console.log("Success. Response is: ", data);
93     var msg = data.message;
94     showResponse(msg);
95   } catch (err) {
96     console.log("Error responding to message.", err);
97   }
98   } catch (err) {
99     console.log("Error translating text.", err);
100   }
101   } catch (err) {
102     console.log("Error identifying language.", err);
103   }
104 }
```

The web browser shows the AWS documentation page for the `SDK 버전 3용 개발자 안내서` (Developer Guide for SDK Version 3). The page title is `export { translateClient };`. The text on the page explains how to update the bot alias and name using the `createResponse` function. A red circle highlights the `BOT_ALIAS` and `BOT_NAME` parameters in the code snippet.

이 코드는 여기에서 확인할 수 있습니다 [GitHub](#).

다음으로, `index.js` 파일을 생성하고 아래 코드를 이 파일에 붙여 넣습니다.

**`BOT_ALIAS` 및 `BOT_NAME`을 각각 Amazon Lex 봇의 별칭 및 이름으로 바꾸고, `USER_ID`를 사용자 ID로 바꿉니다. `createResponse` 비동기 함수는 다음을 수행합니다.**

- 사용자가 입력한 텍스트를 브라우저에 가져오고 Amazon Comprehend를 사용하여 언어 코드를 결정합니다.
- 언어 코드를 가져오고 Amazon Translate를 사용하여 텍스트를 영어로 번역합니다.
- 번역된 텍스트를 가져오고 Amazon Lex를 사용하여 응답을 생성합니다.
- 브라우저 페이지에 응답을 게시합니다.

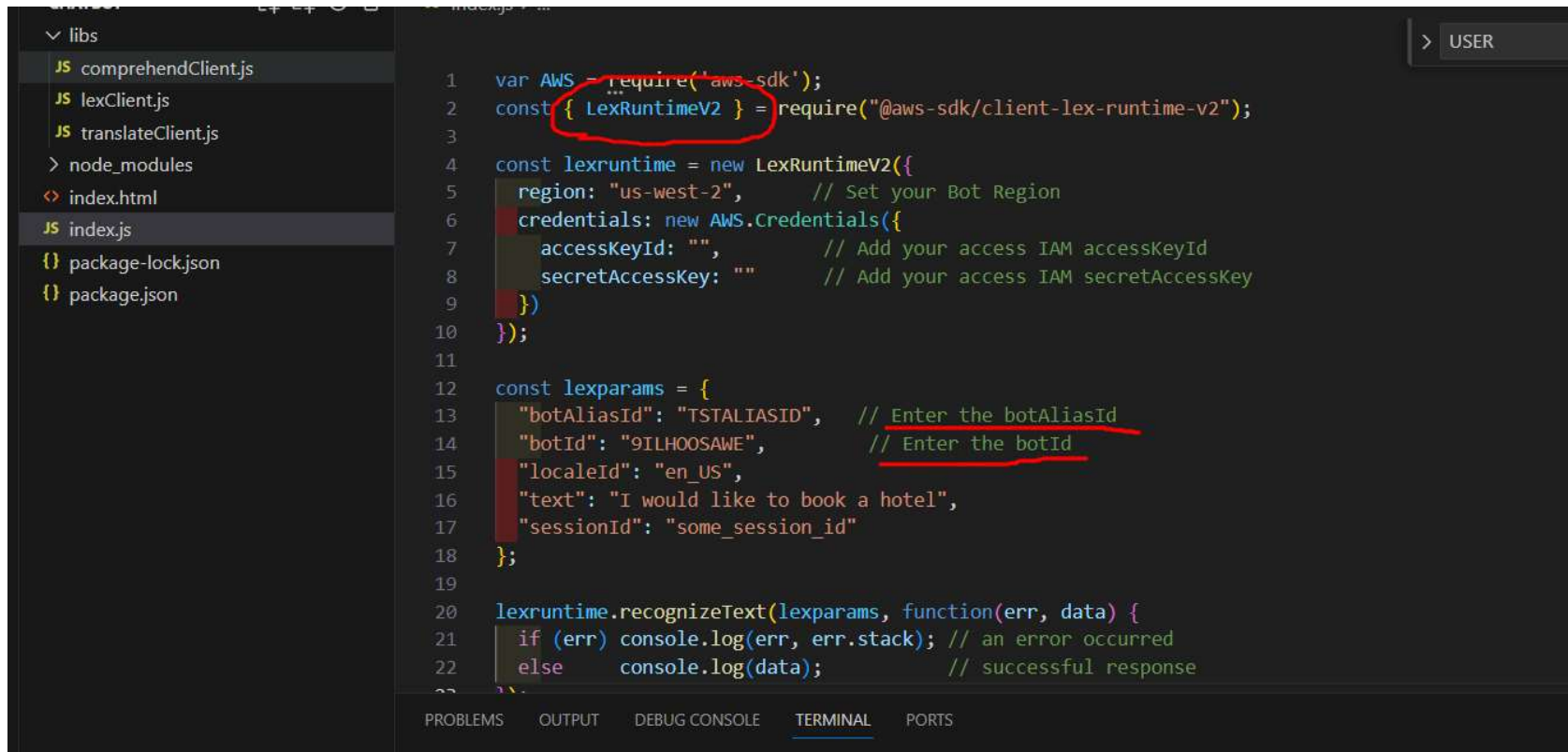
The code snippet at the bottom of the page shows the following imports:

```
import { DetectDominantLanguageCommand } from "@aws-sdk/client-comprehend";
import { TranslateTextCommand } from "@aws-sdk/client-translate";
```

# AccessKey 설정

- `AWS.config.update({`
- `accessKeyId: 'AKIA6GBMC5CPZTSNJUNT',`
- `secretAccessKey:`  
      `'lvWz3qivO/aZAdB+DRjosWR9tUbZaKf2VOObzFwN',`
- `region: 'ap-northeast-2'`
- `});`

# index.js



```
1 var AWS = require('aws-sdk');
2 const { LexRuntimeV2 } = require('@aws-sdk/client-lex-runtime-v2');
3
4 const lexruntime = new LexRuntimeV2({
5   region: "us-west-2", // Set your Bot Region
6   credentials: new AWS.Credentials({
7     accessKeyId: "", // Add your access IAM accessKeyId
8     secretAccessKey: "" // Add your access IAM secretAccessKey
9   })
10 });
11
12 const lexparams = {
13   "botAliasId": "TSTALIASID", // Enter the botAliasId
14   "botId": "9ILHOOSAW", // Enter the botId
15   "localeId": "en_US",
16   "text": "I would like to book a hotel",
17   "sessionId": "some_session_id"
18 };
19
20 lexruntime.recognizeText(lexparams, function(err, data) {
21   if (err) console.log(err, err.stack); // an error occurred
22   else console.log(data); // successful response
23 })
```

cmd창에서 아래 명령어 실행  
npm install @aws-sdk/client-lex-runtime-v2

# IAM 권한추가

Identity and Access Management(IAM)

Q IAM 검색

대시보드

▼ 액세스 관리

사용자 그룹

사용자

역할

정책

자격 증명 공급자

계정 설정

▼ 보고서 액세스

Access Analyzer

외부 액세스

미사용 액세스

보내기 선택

ARN  
arn:aws:iam::975050041503:user/20240328test

생성됨  
March 28, 2024, 20:31 (UTC+09:00)

콘솔 액세스  
비활성화됨

마지막 콘솔 로그인  
-

액세스 키 1  
AKIA6GBMC5CPZTSNJUNT - Active  
사용됨 6일 전. 14일 기존.

액세스 키 2  
액세스 키 만들기

권한

그룹

태그 (1)

보안 자격 증명

액세스 관리자

권한 정책 (2)

사용자에게 직접 연결된 정책을 통해 또는 그룹을 통해 권한을 정의합니다.

검색

필터링 기준 유형  
모든 유형

1

정책 이름

유형

연결 방식

<input type="checkbox"/>	AmazonRekognitionFullAccess	AWS 관리형	직접
<input type="checkbox"/>	AmazonS3FullAccess	AWS 관리형	직접

권한 제거 (선택되지 않음)

# lexfull 권한 추가

## 권한 옵션

☐ 그룹에 사용자 추가

기존 그룹에 사용자를 추가하거나 새 그룹을 생성합니다. 그룹을 사용하여 직무별로 사용자 권한을 관리하는 것이 좋습니다.

☐ 권한 복사

기존 사용자로부터 모든 그룹 멤버십, 연결된 관리형 정책, 인라인 정책 및 기존 권한 경계를 복사합니다.

☒ 직접 정책 연결

관리형 정책을 사용자에게 직접 연결합니다. 사용자에게 연결하는 대신, 정책을 그룹에 연결한 후 사용자를 적절한 그룹에 추가하는 것이 좋습니다.

## 권한 정책 (1184)



Q lex

lexfull



필터링 기준 유형

모든 유형

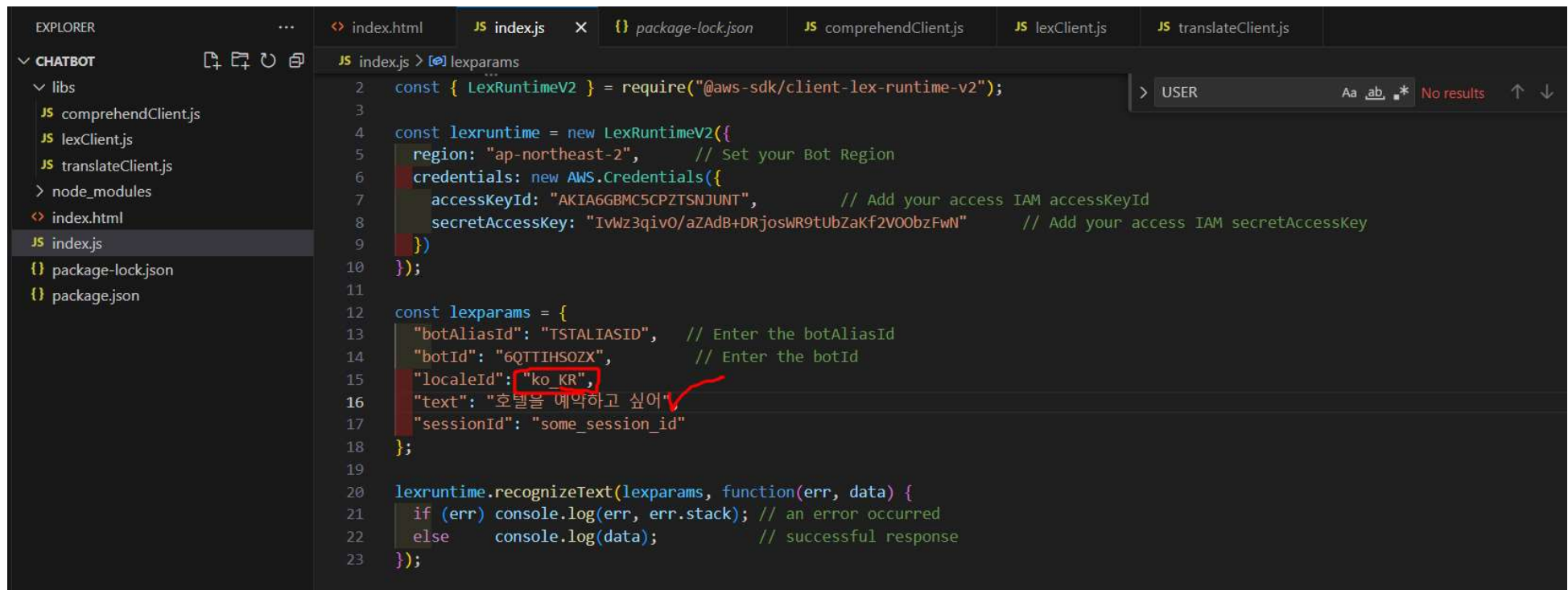
16 개 일치

< 1 >



<input type="checkbox"/>	정책 이름	유형	연결된 엔터티
<input type="checkbox"/>	<a href="#">AlexaForBusinessDeviceSetup</a>	AWS 관리형	0
<input type="checkbox"/>	<a href="#">AlexaForBusinessFullAccess</a>	AWS 관리형	0
<input type="checkbox"/>	<a href="#">AlexaForBusinessGatewayExecution</a>	AWS 관리형	0

# 언어 변경



```
2  const { LexRuntimeV2 } = require("@aws-sdk/client-lex-runtime-v2");
3
4  const lexruntime = new LexRuntimeV2({
5    region: "ap-northeast-2",    // Set your Bot Region
6    credentials: new AWS.Credentials({
7      accessKeyId: "AKIA6GBMC5CPZTSNJUNT",    // Add your access IAM accessKeyId
8      secretAccessKey: "IvWz3qivO/aZAdB+DRjosWR9tUbZaKf2V0ObzFwN"    // Add your access IAM secretAccessKey
9    })
10 });
11
12 const lexparams = {
13   "botAliasId": "TSTALIASID",    // Enter the botAliasId
14   "botId": "6QTTIHSOZX",    // Enter the botId
15   "localeId": "ko_KR",
16   "text": "호텔을 예약하고 싶어",
17   "sessionId": "some_session_id"
18 };
19
20 lexruntime.recognizeText(lexparams, function(err, data) {
21   if (err) console.log(err, err.stack); // an error occurred
22   else console.log(data);    // successful response
23 });
```

# 실행

- cmd창에서 아래 명령어 실행
- node index

```
C:\Windows\System32\cmd x + v
},
interpretations: [
  {
    intent: [Object],
    interpretationSource: 'Lex',
    nluConfidence: [Object]
  },
  {
    intent: [Object],
    interpretationSource: 'Lex',
    nluConfidence: [Object]
  },
  { intent: [Object], interpretationSource: 'Lex' }
],
messages: [ { content: '어떤 도시에서 숙박할 예정입니까?', contentType: 'PlainText' } ],
sessionId: 'some_session_id',
sessionState: {
  dialogAction: { slotToElicit: 'Location', type: 'ElicitSlot' },
  intent: {
    confirmationState: 'None',
    name: 'BookHotel',
    slots: {},
    state: 'InProgress'
  },
  originatingRequestId: '49a7ec74-6af6-40a4-8cef-37ff053a4cb9',
  sessionAttributes: {}
}
}
```