



LangCon 2019



LUIS 머신러닝 자연어처리 기술 기반 챗봇 개발 및 서비스하기

Microsoft AI MVP 강창훈

1. BotFramework 소개 및 개발환경구축

1. Microsoft Bot Framework?
2. 챗봇 개발환경구축

2. 심플챗봇 개발하기

1. 병원 진료예약 챗봇 시연
2. 템플릿기반 심플챗봇개발하기
3. 친절한 챗봇 개발하기
4. 시나리오 기반 챗봇 개발하기

3. Azure BotService로 서비스하기

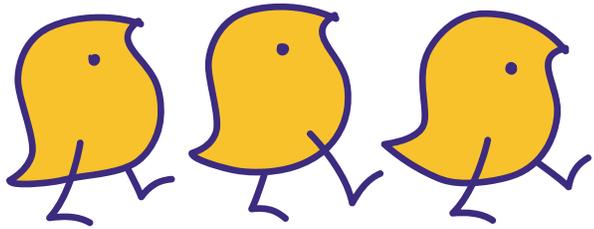
1. ChatBot 개발/서비스 생태계
2. Azure Bot Service?
3. Azure Bot Service 환경
4. Azure Bot Service 구성 및 게시하기

4. Azure Cognitive Service

1. Azure Cognitive Service?
2. Main Cognitive Service
3. 샘플 서비스 소개

5. 머신러닝 자연어처리 LUIS 활용하기

1. LUIS?
2. LUIS Concept
3. LUIS 사용절차
4. LUIS 실습하기



1.BotFramework 소개 및 개발환경구축

1.1 Microsoft Bot Framework?

- 마이크로소프트사의 소프트웨어 기반 봇 개발 프레임워크.
- Bot Framework는 강력하고 인텔리전트 한 봇을 구축하고 연결하며 테스트,배포 가능 환경 제공
- .NET, Node.js 개발환경에서 C#, Javascript 언어를 기본적으로 지원. (Ver 3.x)
- REST 서비스 환경을 지원하여 각종 개발언어를 포괄적으로 지원.
- C#과 Node.js 언어기반 Bot Builder SDK 와 Azure Bot Service를 사용하여 봇을 빠르게 개발 서비스할 수 있음.

<https://dev.botframework.com/>

Tip) Bot framework Ver 4.0 & Bot Builder Ver 4.0 Preview 버전에 대해

- 2018년 02월 새로운 버전의 Botframework Ver4 SDK(Preview)가 2018 Build 행사에서 소개됨
- Ver4.0은 C#, JavaScript, Python 또는 Java를 사용하여 봇을 빌드 할 수 있는 새롭고 풍부한 다중 언어 SDK를 제공.
- 봇 빌더 v4 SDK는 v3 SDK의 진화 된 GitHub의 오픈 소스 프로젝트이며 2018년 하반기 공식 릴리즈 될 예정.

1.2 개발환경 구축 – Visual Studio 2017 설치

1.2.1 Visual Studio 2017 Community Ver 이상 설치

<https://www.visualstudio.com/ko/>

The screenshot shows the Visual Studio 2017 installation interface. The title bar reads '설치 — Visual Studio Enterprise 2017 — 15.2(26430.12)'. Below the title bar, there are three tabs: '워크로드' (Workloads), '개별 구성 요소' (Individual components), and '언어 팩' (Language packs). The '워크로드' tab is selected, showing a list of workloads under the heading 'Windows(3)'. Three workloads are listed: '유니버설 Windows 플랫폼 개발' (Universal Windows Platform development), '.NET 데스크톱 개발' (.NET Desktop development), and 'C++를 사용하여 데스크톱 개발' (Desktop development with C++). Below this, there is a section for '웹 및 클라우드(7)' (Web and Cloud) with a red border. Two workloads are selected with checkmarks: 'ASP.NET 및 웹 개발' (ASP.NET and Web development) and 'Azure 개발' (Azure development). Below this, there are two more workloads: 'Python 개발' (Python development) and 'Node.js 개발' (Node.js development).

설치 — Visual Studio Enterprise 2017 — 15.2(26430.12)

워크로드 개별 구성 요소 언어 팩

Windows(3)

유니버설 Windows 플랫폼 개발
C#, VB, JavaScript 또는 원한다면 C++으로 유니버설 Windows 플랫폼용 응용 프로그램을 만들어 보세요.

.NET 데스크톱 개발
.NET Framework를 사용하여 WPF, Windows Forms 및 콘솔 응용 프로그램을 빌드해 보세요.

C++를 사용하여 데스크톱 개발
강력한 Visual C++ 도구 집합을 사용하여 클래식 Windows 기반 응용 프로그램을 빌드해 보세요. 또는 원하신다면 MFC 같은 기능...

웹 및 클라우드(7)

ASP.NET 및 웹 개발
ASP.NET, ASP.NET Core, HTML, JavaScript 및 CSS를 사용하여 웹 응용 프로그램을 빌드해 보세요.

Azure 개발
클라우드 앱을 개발하고 리소스를 만들기 위한 Azure SDK, 도구 및 프로젝트를 다룹니다.

Python 개발
Python에 대한 편집, 디버깅, 대화형 개발 및 원본 제어를 다룹니다.

Node.js 개발
비동기 이벤트 중심 JavaScript 런타임인 Node.js를 사용하여 확장성 있는 네트워크 응용 프로그램을 빌드해 보세요.

1.2 개발환경 구축 – Bot 개발 템플릿 설치

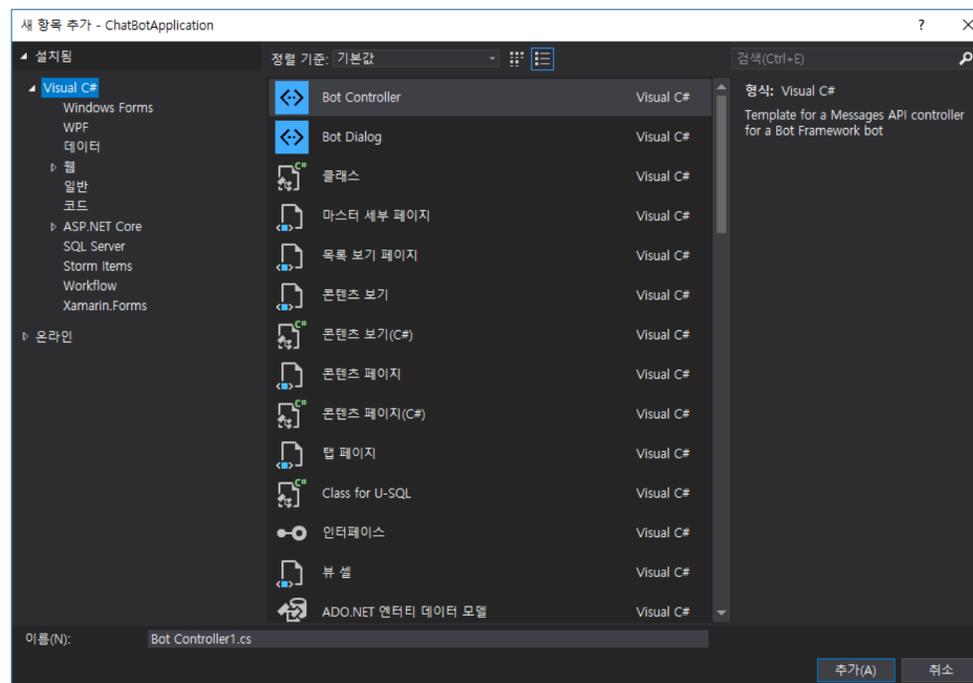
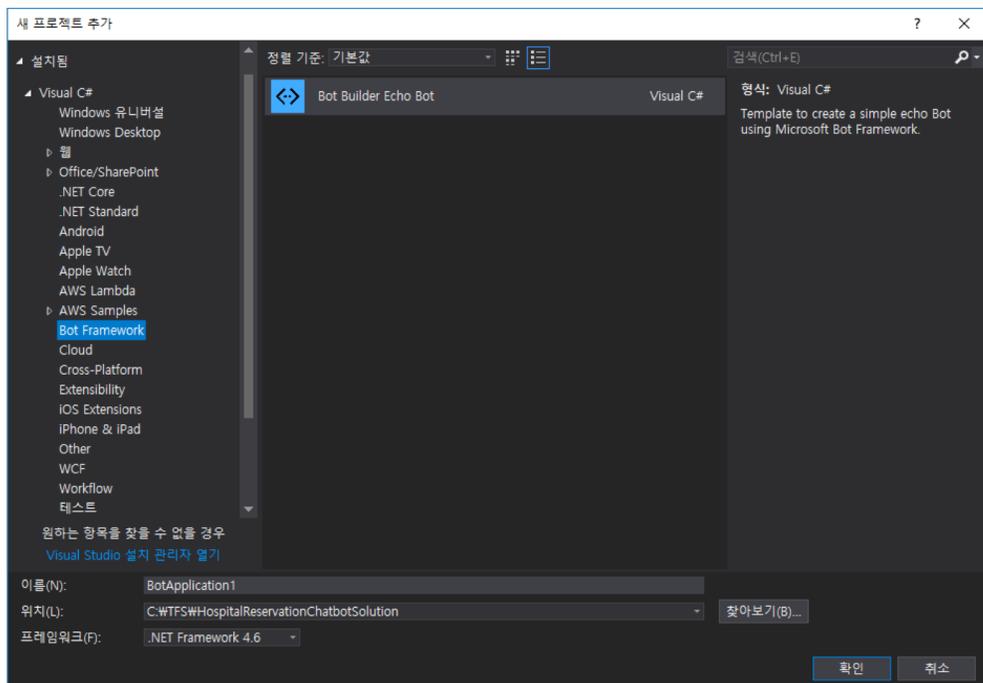
1.2.2 Bot 개발 템플릿 설치

<https://docs.microsoft.com/en-us/azure/bot-service/?view=azure-bot-service-3.0>

<http://aka.ms/bf-bc-vstemplate>

<http://aka.ms/bf-bc-vscontrollertemplate>

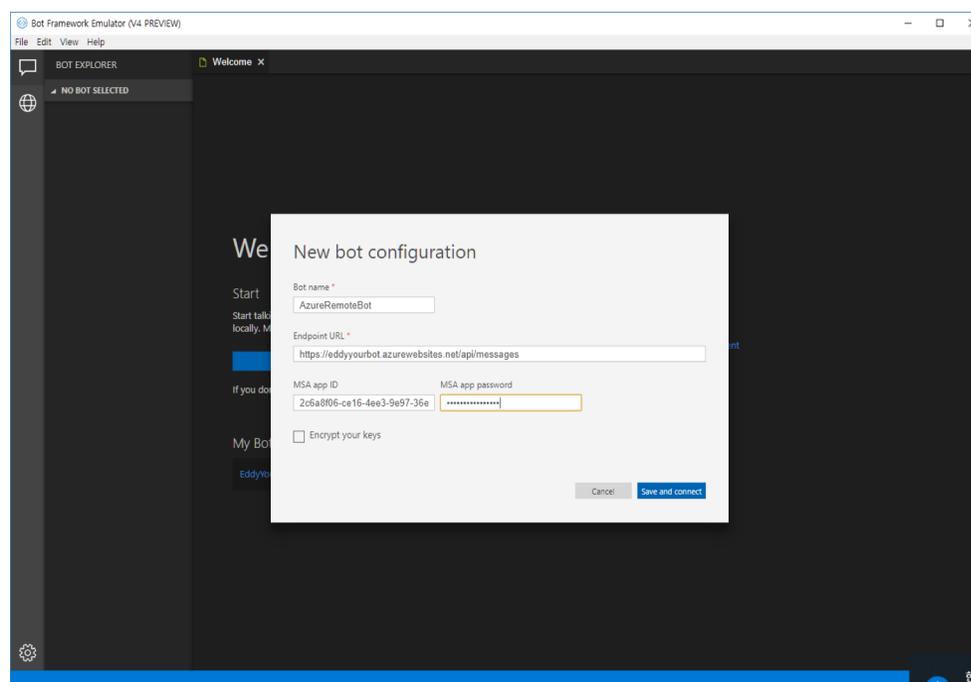
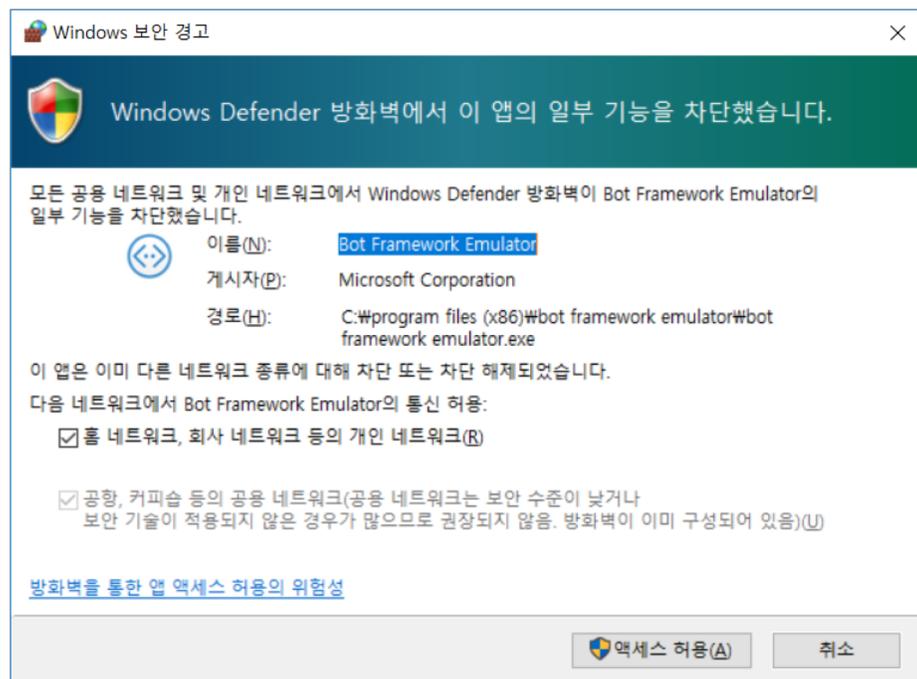
<http://aka.ms/bf-bc-vsdialogtemplate>



1.2 개발환경 구축 – Bot Emulator 설치

1.2.3 Bot Emulator 설치

<https://github.com/Microsoft/BotFramework-Emulator/releases>



1. 병원 진료예약 챗봇 시연

회원이시면 예를 그렇지 않으면 아니오 를 입력해주세요.

Bot

예
User



서비스선택
아래 원하시는 서비스유형을 선택해주세요.

- 진료예약
- 진료안내
- 오시는길
- 개인정보변경
- 병원소개

Bot 오전 7:46:37

메세지를 입력하세요...

2. 템플릿기반 심플챗봇개발하기

The image shows a development environment for a chatbot. On the left, Visual Studio displays the code for `ChatBotApplication.MessagesController`. The code includes using statements for `System.Net`, `System.Threading.Tasks`, `System.Web.Http`, `Microsoft.Bot.Builder.Dialogs`, and `Microsoft.Bot.Connector`. It defines a `MessagesController` class with a `Post` method that handles incoming messages and sends a response.

In the center, a browser window at `localhost:3978` displays the `ChatBotApplication` landing page. The page title is `ChatBotApplication` and the content reads: "This is a simple echo bot sample using Microsoft Bot Framework. Visit [Bot Framework](#) to register your bot. When you register it, remember to set your bot's endpoint to `https://your_bot_hostname/api/messages`". There is a `Deploy to Azure` button and a `Prerequisites` section.

On the right, the Bot Framework Emulator is open, showing a chat window with the text "You sent 하이 which was 2 characters" and a response from the bot: "bot 오후 9:08:41". The emulator interface includes a `TRANSCRIPT EXPLORER` on the left and a `INSPECTOR` on the right. The `INSPECTOR` shows a log of events, including the bot listening on `http://localhost:53341` and `https://b0ea6a66.ngrok.io`, and the receipt of a `POST 201` request with the message "하이".

3. 친절한 챗봇 개발하기

The screenshot displays the Bot Framework Emulator interface for a 'ChatBotApplication'. The main chat window shows a conversation where the bot greets the user and provides information about a hospital. The user asks for an example, and the bot asks for a member ID. The user provides 'Gentleman', and the bot asks for a phone number. The user enters '01027605246'. The right-hand 'INSPECTOR' pane shows a 'LOG' of the underlying API calls, including messages and POST requests to the directline API.

Chat Transcript:

- Bot: 안녕하세요. 한국 대학교 병원 진료예약 도우미 챗봇입니다.
- User: 하이
- Bot: 회원이시면 예를 그렇지 않으면 아니오 를 입력해주세요.
- User: 아니오
- Bot: 신규 회원 가입 프로세스를 진행합니다. 회원 아이디를 입력해주세요.
- User: Gentleman
- Bot: 전화번호를 입력해주세요.
- User: 01027605246

LOG:

```
[07:37:20] Emulator listening on http://localhost:56918
[07:37:20] ngrok listening on https://b4c33117.ngrok.io
[07:37:20] ngrok traffic inspector: http://127.0.0.1:4040
[07:37:20] Will bypass ngrok for local addresses
[07:37:20] POST 201 directline.startConversation
[07:37:20] POST 200 conversations.replyToActivity
[07:37:20] <- message 안녕하세요. 한국 대학교 병원 진료예약 도우미 챗봇입니다.
[07:37:24] -> message 하이
[07:37:24] POST 200 conversations.replyToActivity
[07:37:24] <- message 회원이시면 예를 그렇지 않으면 아니오 를 입력해주세요.
[07:37:24] POST 200 directline.postActivity
[07:37:27] -> message 아니오
[07:37:27] POST 200 conversations.replyToActivity
[07:37:27] <- message 신규 회원 가입 프로세스를 진행합니다. 회원 아이디를 입력해주세요.
[07:37:27] POST 200 directline.postActivity
[07:37:43] -> message Gentleman
[07:37:43] POST 200 conversations.replyToActivity
[07:37:43] <- message 전화번호를 입력해주세요.
[07:37:44] POST 200 directline.postActivity
```

4. 시나리오 기반 챗봇 개발하기

The screenshot displays the Bot Framework Emulator (V4 PREVIEW) interface for a chatbot application. The main chat window shows a bot message: "회원하시면 예를 그렇지 않으면 아니오 를 입력해주세요." Below this is a card titled "서비스선택" (Service Selection) with the instruction "아래 원하시는 서비스유형을 선택해주세요." (Please select the service type you want below). The card contains five buttons: "진료예약" (Appointment), "진료안내" (Consultation), "오시는길" (Directions), "개인정보변경" (Change Personal Information), and "병원소개" (Hospital Introduction). A user has responded with "예" (Yes), which is shown in a yellow bubble. The "INSPECTOR - JSON" panel on the right shows the incoming message payload:

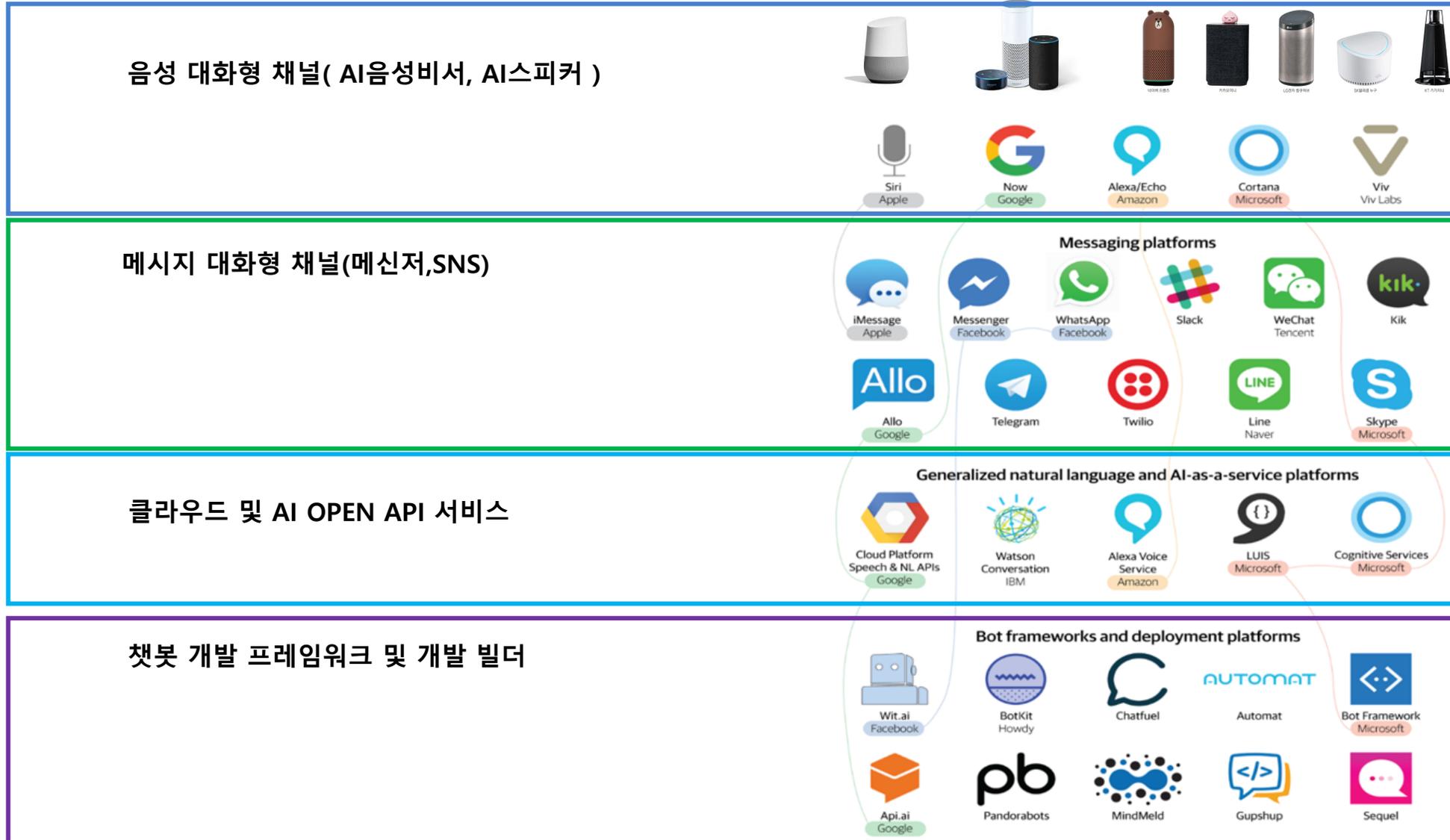
```
{
  "channelData": {
    "clientActivityId": "1534891585366.23038554508555298.2"
  },
  "from": {
    "id": "default-user",
    "name": "User"
  },
  "id": "1007f560-a594-11e8-82d4-2b39181231d7",
  "locale": "ko",
  "text": "예",
  "textFormat": "plain",
  "timestamp": "2018-08-21T22:46:37.749Z",
  "type": "message"
}
```

The "LOG" panel at the bottom shows the emulator's activity, including the user's message and the bot's response:

```
[07:46:25] Emulator listening on http://localhost:57175
[07:46:25] ngrok listening on https://01974457.ngrok.io
[07:46:25] ngrok traffic inspector: http://127.0.0.1:4040
[07:46:25] Will bypass ngrok for local addresses
[07:46:25] POST 201 directline.startConversation
[07:46:30] POST 200 conversations.replyToActivity
[07:46:30] <- message 안녕하세요. 한국 대학교 병원 진료예약 도우미 챗봇입니다.
[07:46:35] -> message 하이
[07:46:35] POST 200 conversations.replyToActivity
[07:46:35] <- message 회원하시면 예를 그렇지 않으면 아니오 를 입력해주세요.
[07:46:35] POST 200 directline.postActivity
[07:46:37] -> message 예
[07:46:37] POST 200 conversations.replyToActivity
[07:46:37] <- message application/vnd.microsoft.card.hero
[07:46:37] POST 200 directline.postActivity
```

3. Azure Bot Service로 서비스하기

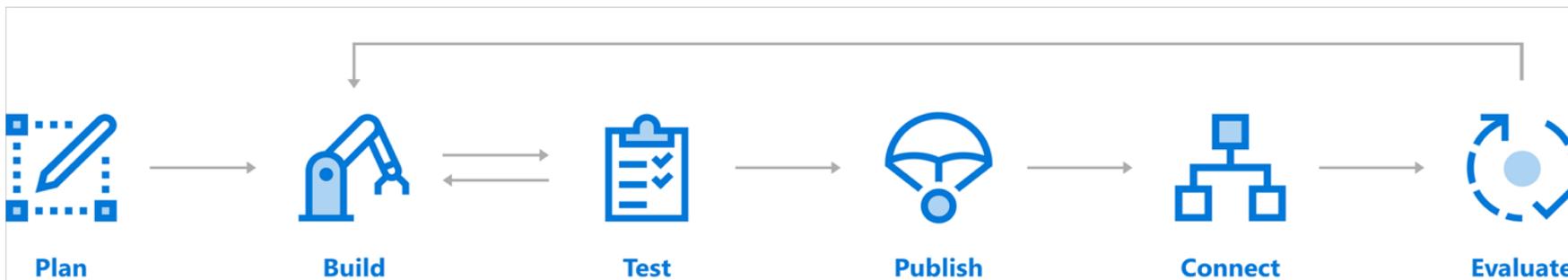
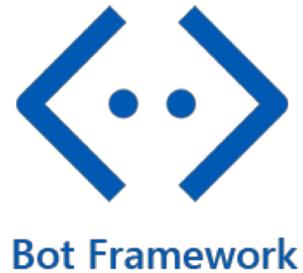
1. ChatBot 개발/서비스 생태계



3. Azure Bot Service로 서비스하기

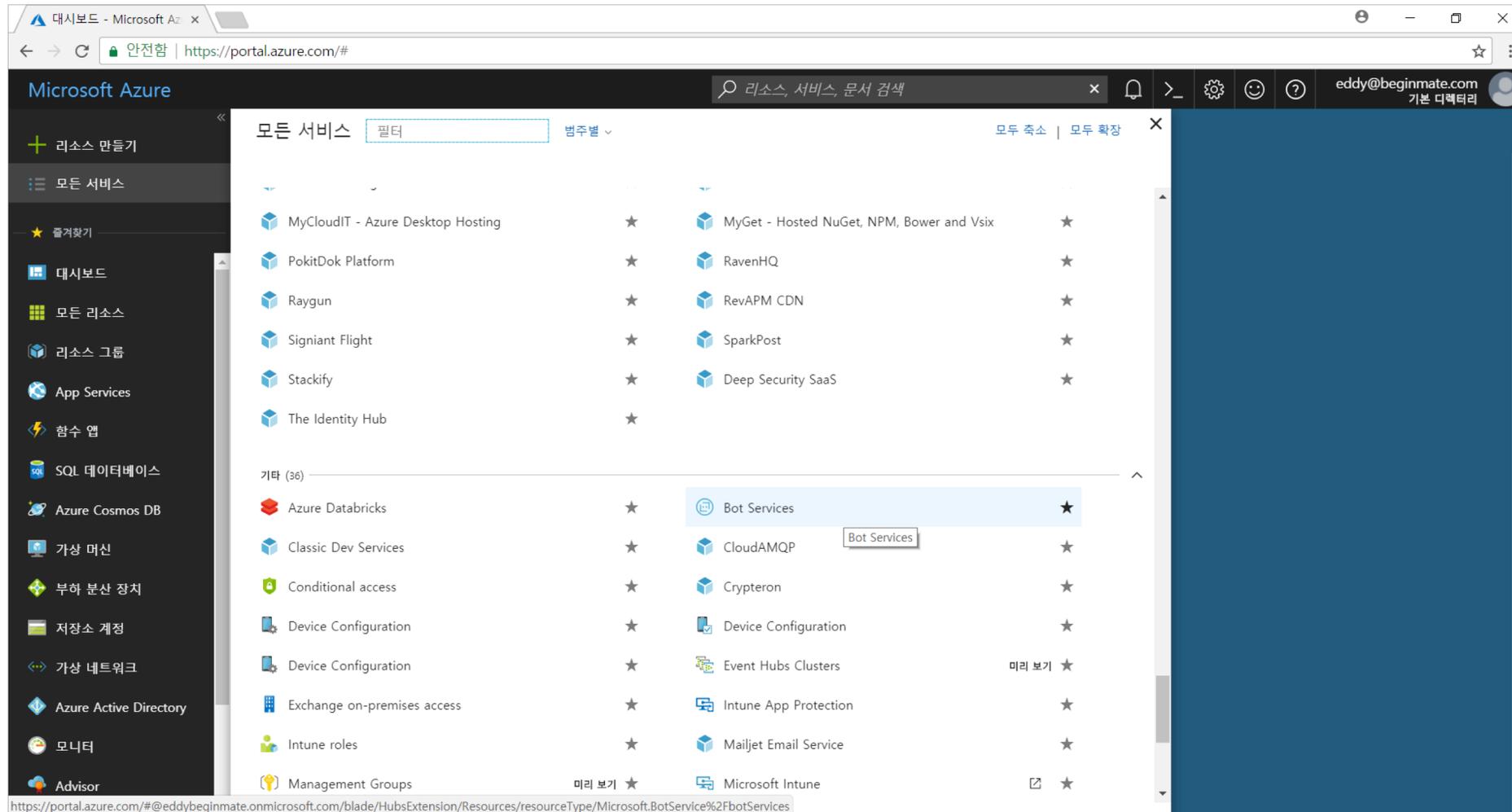
2. Azure Bot Service?

- Azure Bot Service On Azure Cloud
- Bot Builder ver3, ver4 & Visual Studio 2017 On Locally



3. Azure Bot Service로 서비스하기

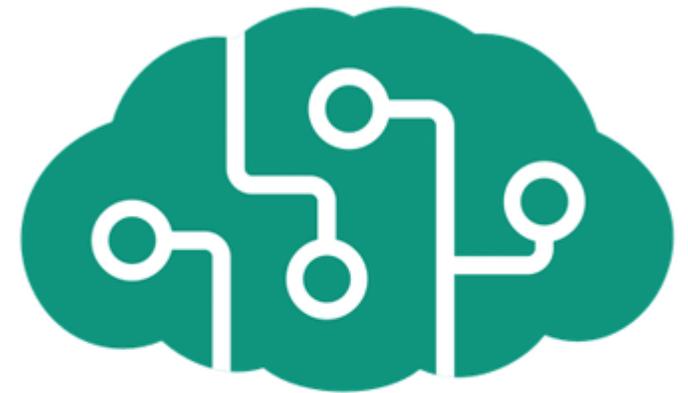
4. Azure Bot Service 구성 및 게시하기



1. Azure Cognitive Service?

“**각종 클라우드 기반 AI 기능을 쉽게 활용해 비즈니스 문제 해결을 돕는 AZURE AI OPEN API 서비스**”

각종 응용 프로그램, 웹사이트 및 봇에 사용자의 요구 사항을 자연스러운 의사 소통 방식으로 보고(Vision) 듣고(Listen) 말하고(Speak) 이해(Understanding)하고 해석하는
지능적인 알고리즘 제공 서비스



 Microsoft
Cognitive Services

2. Main Cognitive Service

1) 시각

- Computer Vision API : 이미지분석, 텍스트분석, 유명인/랜드마크인식/실시간비디오분석
- Face API : 얼굴감지, 사람식별, 감정인식

2) 음성

- Speaker Recognition API
- 음성 서비스 : 음성기반 화자식별, 음성TO텍스트, 텍스트To음성, 음성번역(통역), 한국어 미지원

3) 언어

- LUIS(Language Understanding) : 자연어처리
- Translation API : 언어간 기계번역
- Text Analytics API: 텍스트 분석, 언어, 감정, 핵심 문구 및 엔터티 – 한국어 미지원

4) 지식

- QnA Maker API : 질문 답하기

5) 검색

- Bing Search : Bing기반의 각종 데이터 검색 기능제공

3. 샘플 서비스 소개

이미지 분석

이 기능은 이미지에서 찾은 시각적 콘텐츠에 대한 정보를 제공합니다. 태그 지정, 도메인 특정 모델 및 설명에 대해 4개의 언어를 사용하여 자신 있게 콘텐츠를 파악하고 레이블을 지정하실 수 있습니다. 잠재적인 성인 콘텐츠를 감지할 수 있도록 성인/외설 설정을 적용합니다. 사진의 이미지 형식과 색 구성표를 파악합니다.

실제 작동되는 방식을 확인해보세요.



기능 이름:	
설명	<pre>{ "tags": ["train", "platform", "station", "building", "indoor", "subway", "track", "walking", "waiting", "pulling", "board", "people", "man", "luggage", "standing", "holding", "large", "woman", "yellow", "suitcase"], "captions": [{ "text": "people waiting at a train station", "confidence": 0.8330993 }] }</pre>
태그	<pre>[{ "name": "train", "confidence": 0.9975446 }, { "name": "platform", "confidence": 0.995543063 }, { "name": "station", "confidence": 0.9798007 }, { "name": "indoor", "confidence": 0.9277198 }, { "name": "subway", "confidence": 0.838939548 }, { "name": "pulling", "confidence": 0.4317156 }]</pre>
이미지 지	"jpeg"

이미지 URL

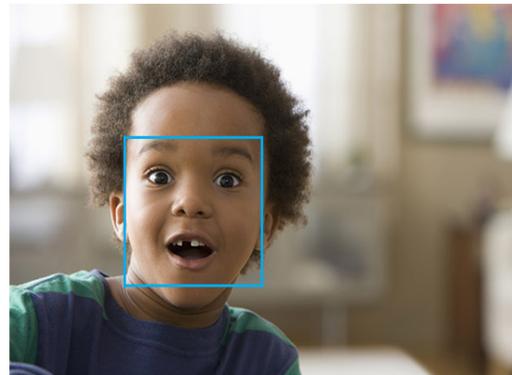
제출

🔍 찾아보기

감정 인식

Face API는 이제 감정 인식을 통합하여, 이미지에 있는 각 얼굴의 분노, 경멸, 혐오, 공포, 행복, 무표정, 슬픔, 놀람 같은 일련의 감정을 확실하게 인식합니다. 이러한 감정은 특정 얼굴 표정으로 서로 다른 문화에서 보편적으로 그 의미가 전달됩니다.

실제 작동되는 방식을 확인해보세요.



감지 결과:
1개 얼굴이 감지됨

```
JSON:  
{  
  {  
    "faceRectangle": {  
      "top": 141,  
      "left": 130,  
      "width": 162,  
      "height": 162  
    },  
    "scores": {  
      "anger": 9.29041E-06,  
      "contempt": 0.000118981574,  
      "disgust": 3.15619363E-05,  
      "fear": 0.000589638,  
      "happiness": 0.06630674,  
      "neutral": 0.00555004273,  
      "sadness": 7.44669524E-06,  
      "surprise": 0.9273863  
    }  
  }  
}
```

이미지 URL

제출

🔍 찾아보기

1. LUIS?

- LUIS(Language Understanding)는 사용자 지정 기계 학습 인텔리전스를 사용자 메시지(음성, 텍스트)로부터 전체적인 의미를 예측하고 관련된 자세한 정보를 추출하는 클라우드 기반 자연어 처리 API 서비스
- 응용 프로그램에서 사람의 말을 통해 무엇을 원하는지 이해(인텐트=의도파악)할 수 있다.
- LUIS 앱은 클라이언트 응용 프로그램이 현명한 선택을 내릴 수 있도록 인텔리전스(지능)를 제공한다.
- LUIS는 기계 학습을 사용하여 사용자 입력(음성, 문자)를 자연어로 받아서 의미를 추출하는 각종 응용 프로그램을 개발자가 쉽게 개발하고 서비스할 수 있는 환경을 제공한다.

<https://www.luis.ai>

<https://azure.microsoft.com/ko-kr/services/cognitive-services/>

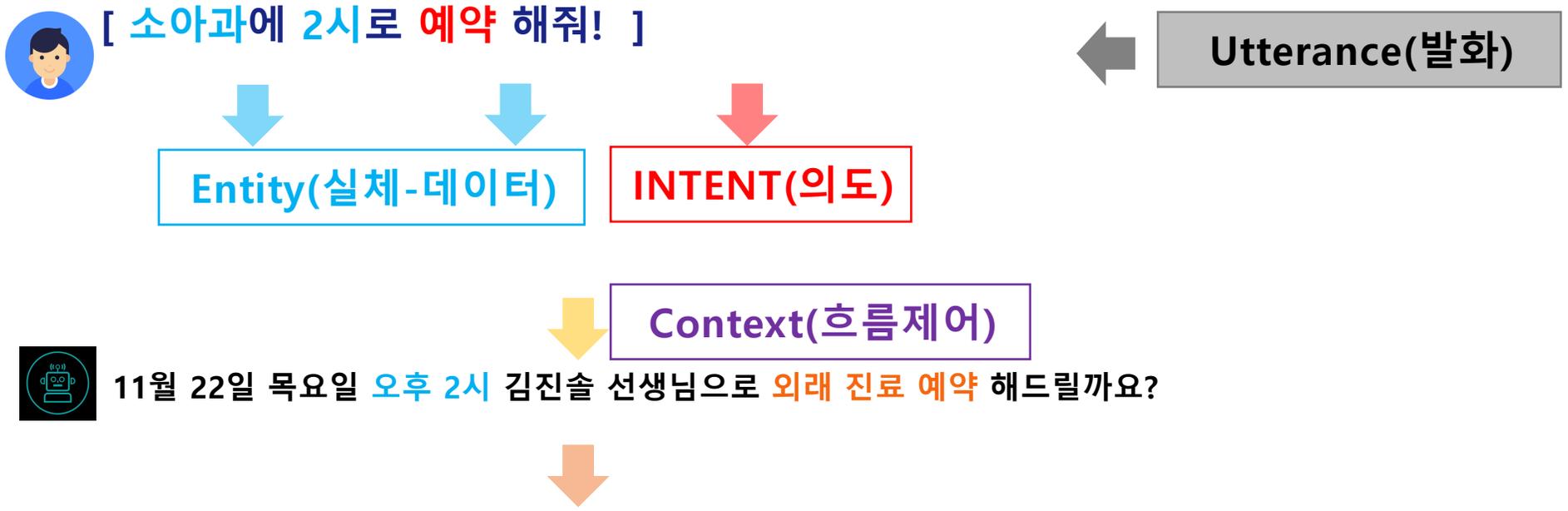
<https://docs.microsoft.com/ko-kr/azure/cognitive-services/luis/>



5. 머신러닝 자연어처리 LUIS 활용하기

2. LUIS Concept : ML기반 NLP 핵심요소

- 목표: 사용자 메시지(텍스트)를 컴퓨터 프로그램이 이해할 수 있는 것으로 번역 하는 것(자연어처리)
- 방법: 사용자 메시지(발화=Utterance) 텍스트를 의도(Intent) 와 실체(Entity)로 번역 한다.
- 발화(Utterance) : 사용자 메시지 또는 음성
- 의도(Intent) : 사용자가 이루고자 하는 목표,메소드(실행목표)
- 실체(Entity) : 의도를 이루기 위한 행위(메소드)에 전달되는 매개변수로 사용자 메시지내의 사실 또는 데이터.
- 컨텍스트(Context) : 대화 흐름/문맥 관리



3. LUIS 사용절차

STEP1. 자연어 처리 모델 만들기

STEP2. Intent 생성-발화등록

STEP3. Entity 등록 및 발화 맵핑

STEP4. 훈련시키기: ML Train

STEP5. 배포하기 : Publish as Open API

4. LUIS 실습하기

LUIS 실습하기
DEMO