

# 이제는 챗봇 시대! 인텔전트 챗봇 개발의 모든 것

## The age of Chatbot

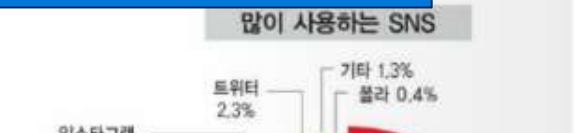
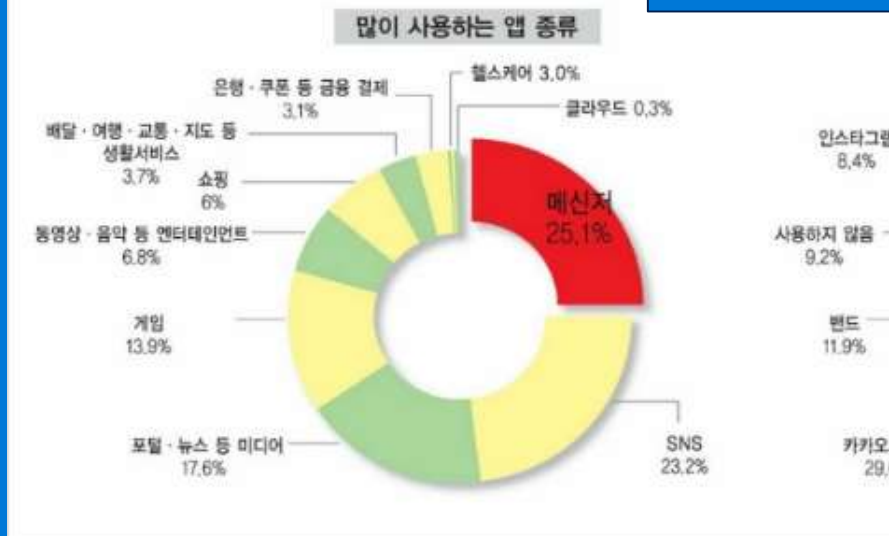
김영욱 부장

Technical Evangelist / DX

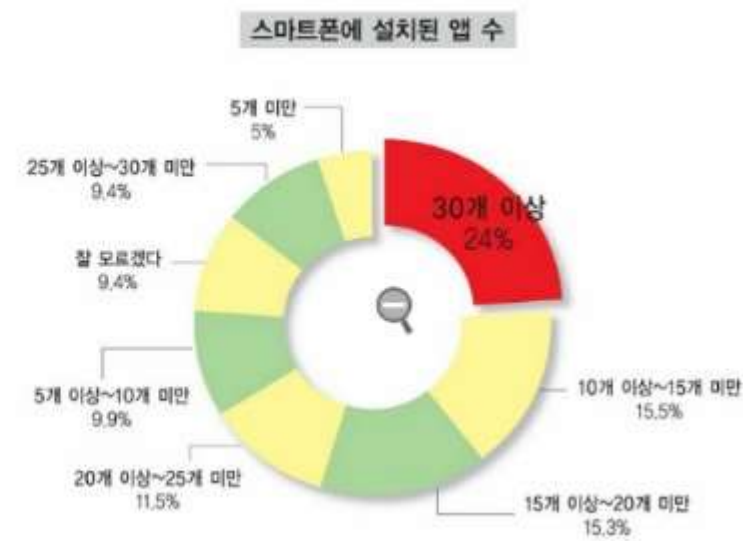
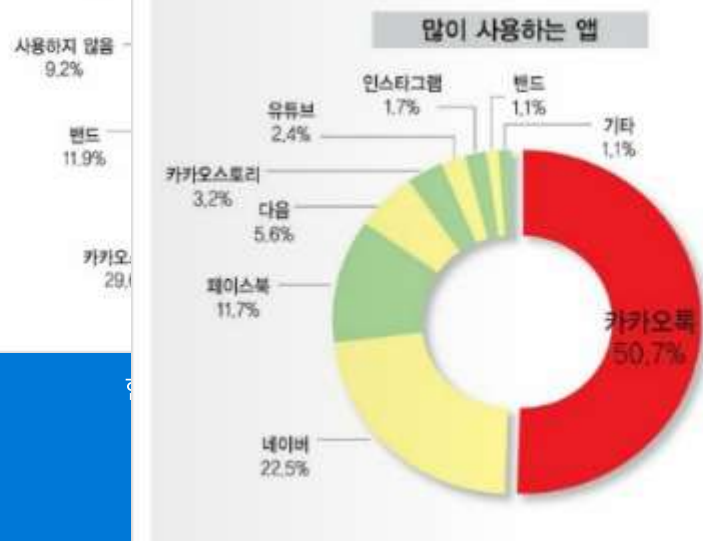
Blog://YoungWook.com

youngwook@outlook.com

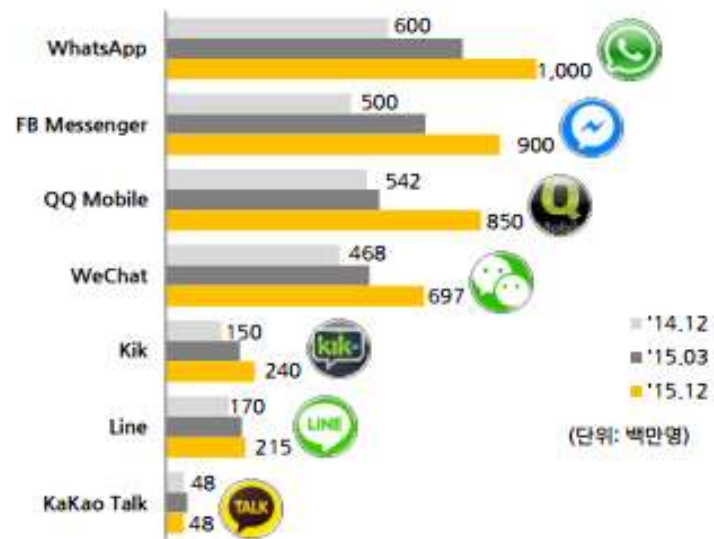
메신저 앱 25.1%



카카오톡 50.7%



[그림1] 주요 모바일 메신저 이용자 수(MAU)



자료: Digieco 및 각사 발표자료 업데이트

[그림2] 4대 SNS vs. 메신저 이용자 수



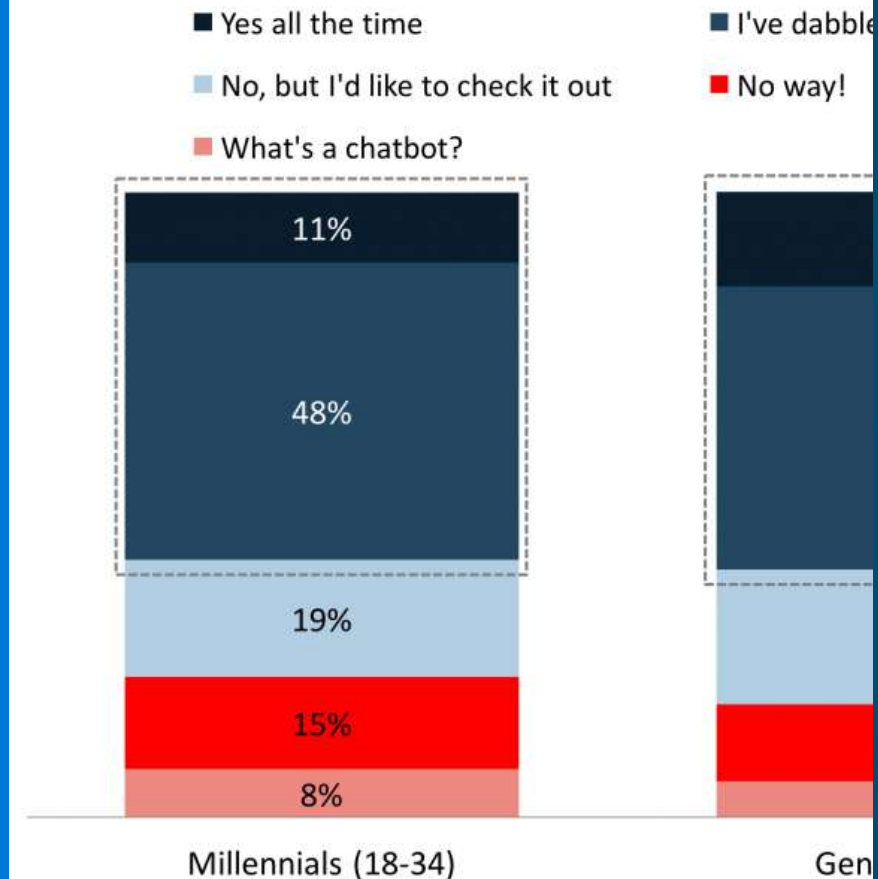
주: 2015년 3분기 기준 (이용자 수 단순 합계)  
자료: Business Insider (2015.11)



대화하는 인공지능, 지금은 챗봇시대!

## US Chatbot Use Among Millennials And Gen Xers

Question: Have you ever interacted with a chatbot over a messaging app?



Source: BI Intelligence survey (n=1,107), October 2016

## CHATBOT ECOSYSTEM

### Deployment Channels



### Third-party Chatbots



### Enabling Technology



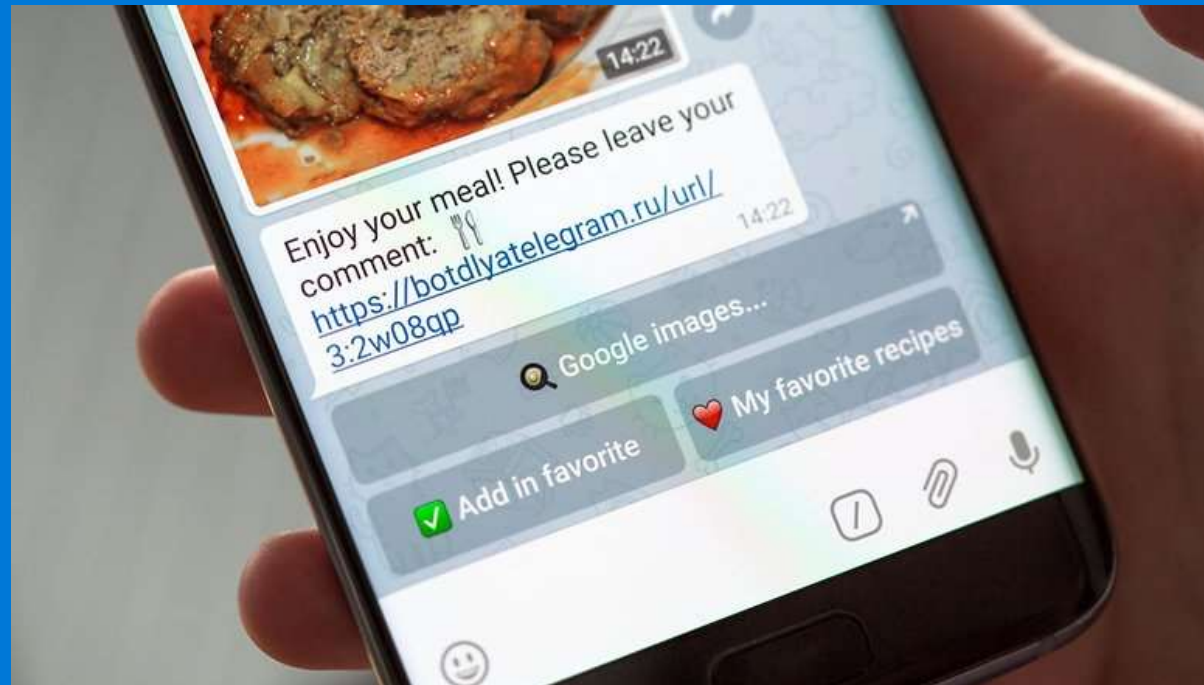
### Native

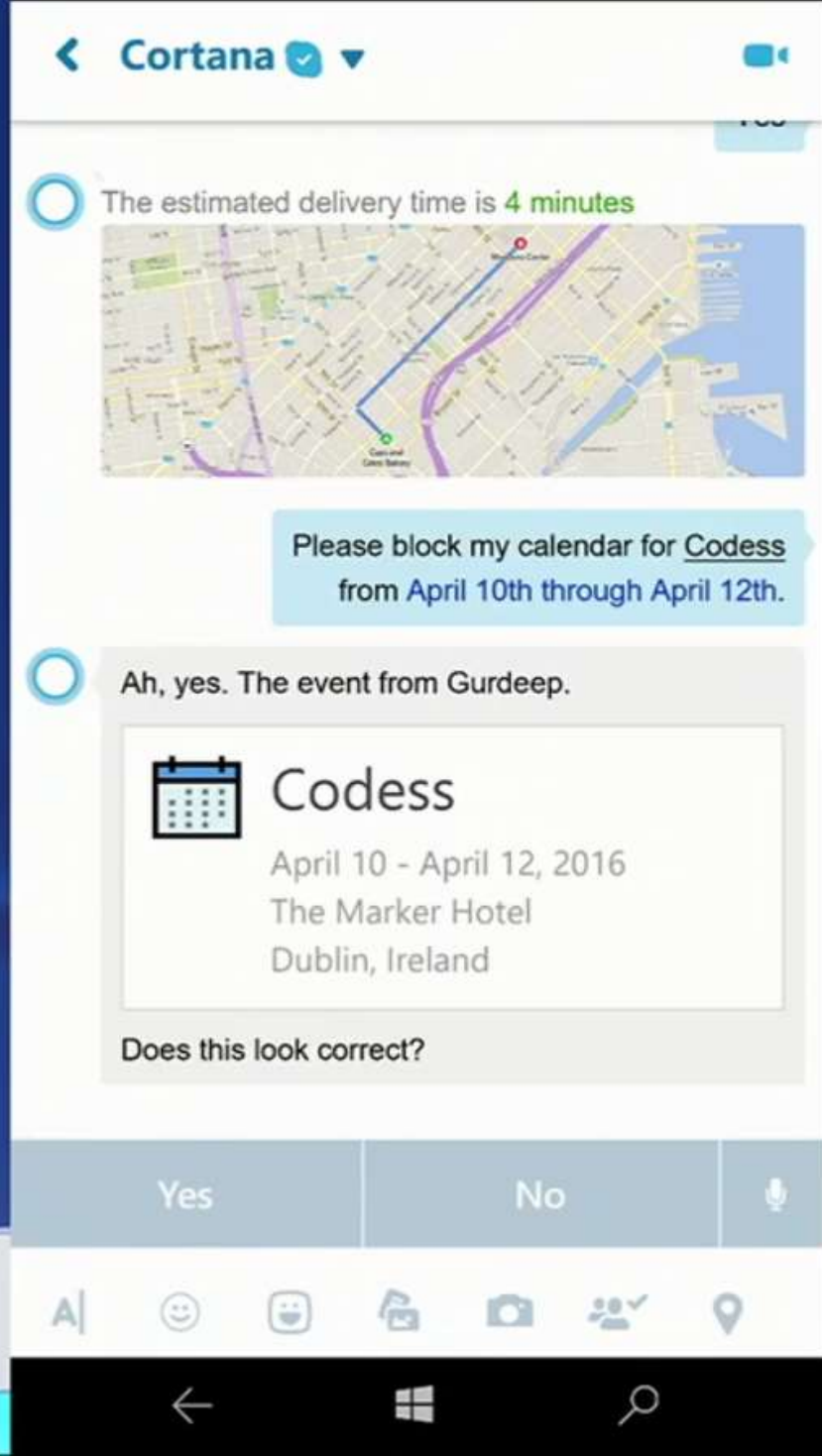


BI INTELLIGENCE



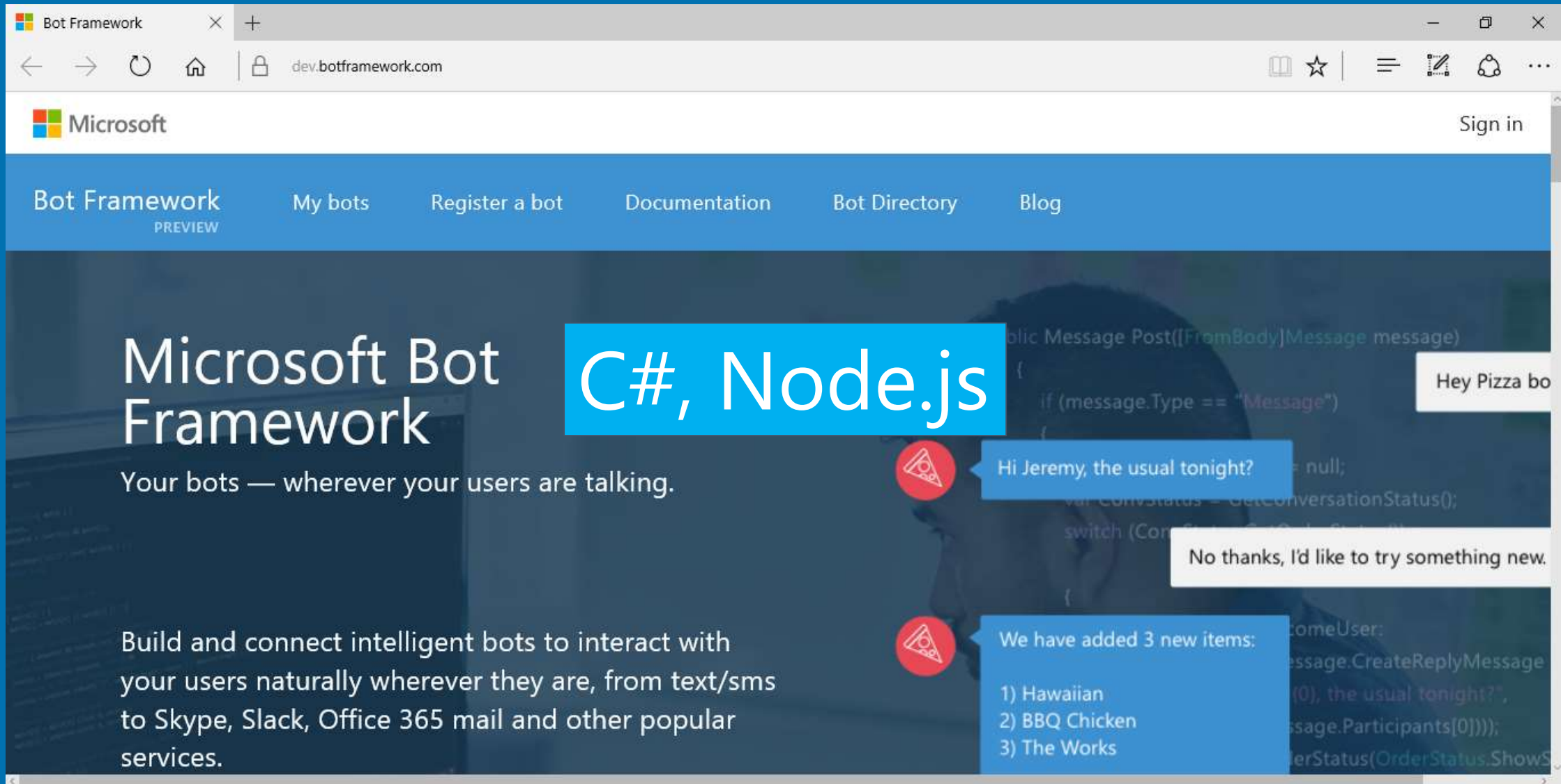
# Microsoft Chatbot.





# MS Bot Framework

<https://dev.botframework.com>



The image is a screenshot of the Microsoft Bot Framework website. The browser's address bar shows 'dev.botframework.com'. The website's header includes the Microsoft logo, a 'Sign in' button, and a navigation menu with links: 'Bot Framework PREVIEW', 'My bots', 'Register a bot', 'Documentation', 'Bot Directory', and 'Blog'. The main content area features the heading 'Microsoft Bot Framework' and the tagline 'Your bots — wherever your users are talking.' Below this, it states: 'Build and connect intelligent bots to interact with your users naturally wherever they are, from text/sms to Skype, Slack, Office 365 mail and other popular services.' A blue rectangular overlay with the text 'C#, Node.js' is positioned over the right side of the main content. In the background, there is a faint image of a person's face and a chat interface showing a conversation with a bot named 'Hey Pizza bot'. The chat messages are: 'Hi Jeremy, the usual tonight?' (in a blue bubble) and 'No thanks, I'd like to try something new.' (in a white bubble). Below the first message, a list of items is shown: 'We have added 3 new items: 1) Hawaiian, 2) BBQ Chicken, 3) The Works'.

Bot Framework  
PREVIEW

My bots Register a bot Documentation Bot Directory Blog

## Microsoft Bot Framework

Your bots — wherever your users are talking.

Build and connect intelligent bots to interact with your users naturally wherever they are, from text/sms to Skype, Slack, Office 365 mail and other popular services.

### C#, Node.js

Hey Pizza bot

Hi Jeremy, the usual tonight?

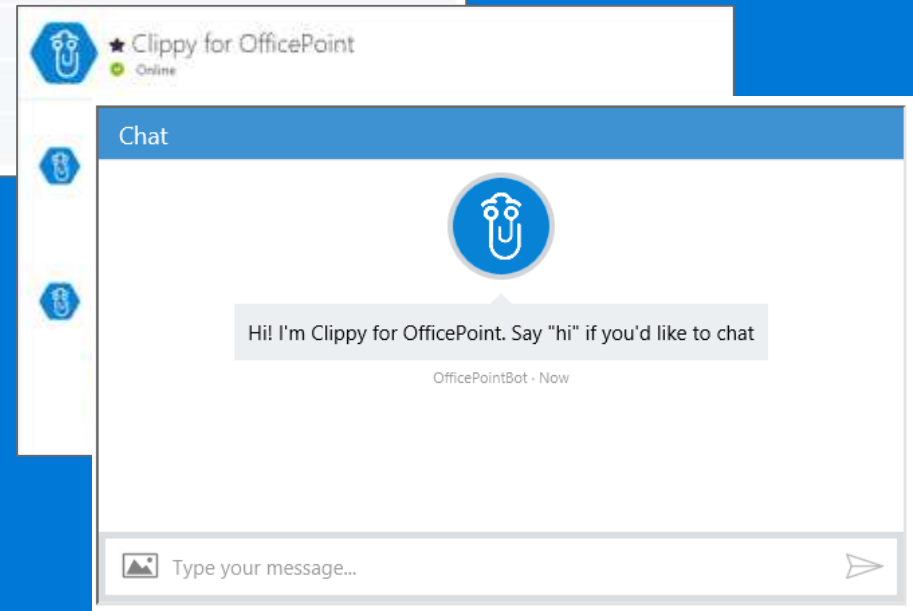
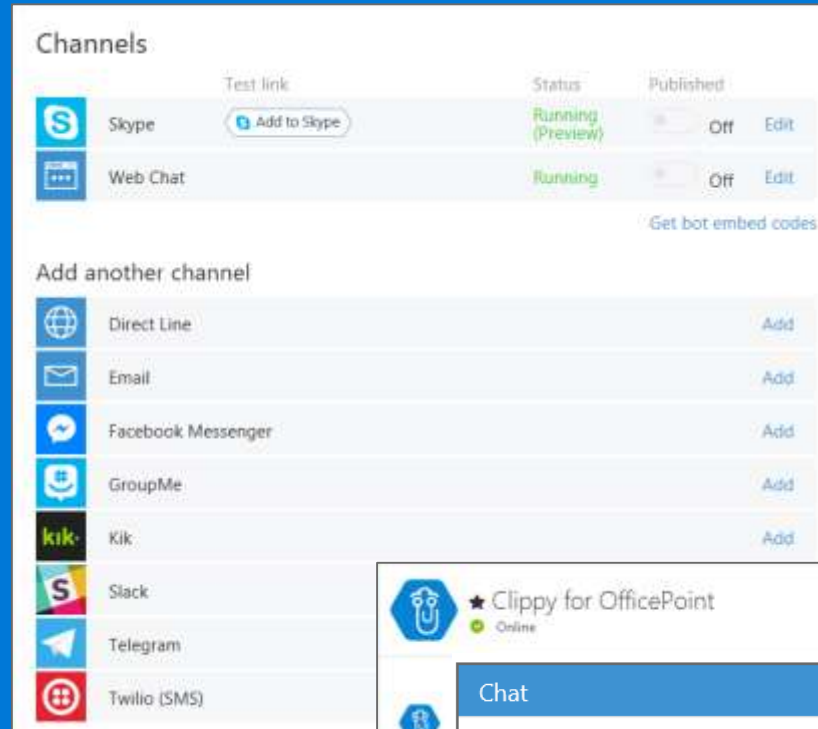
No thanks, I'd like to try something new.

We have added 3 new items:

- 1) Hawaiian
- 2) BBQ Chicken
- 3) The Works

# MS Bot Framework

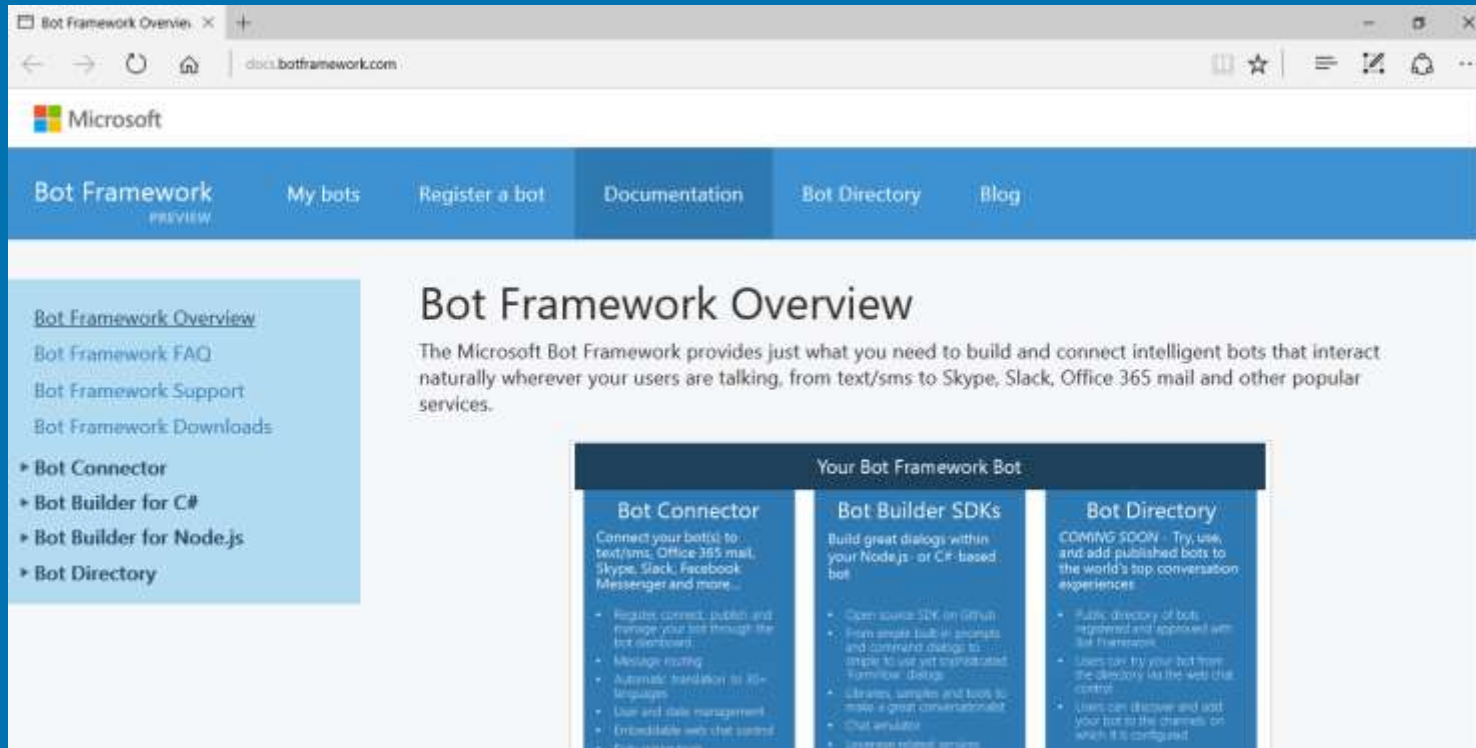
Skype,  
Web, email,  
Facebook,  
GroupMe,  
Kik,  
Slack,  
Telegram,  
Twilio,  
direct line app  
integration.





# Overview

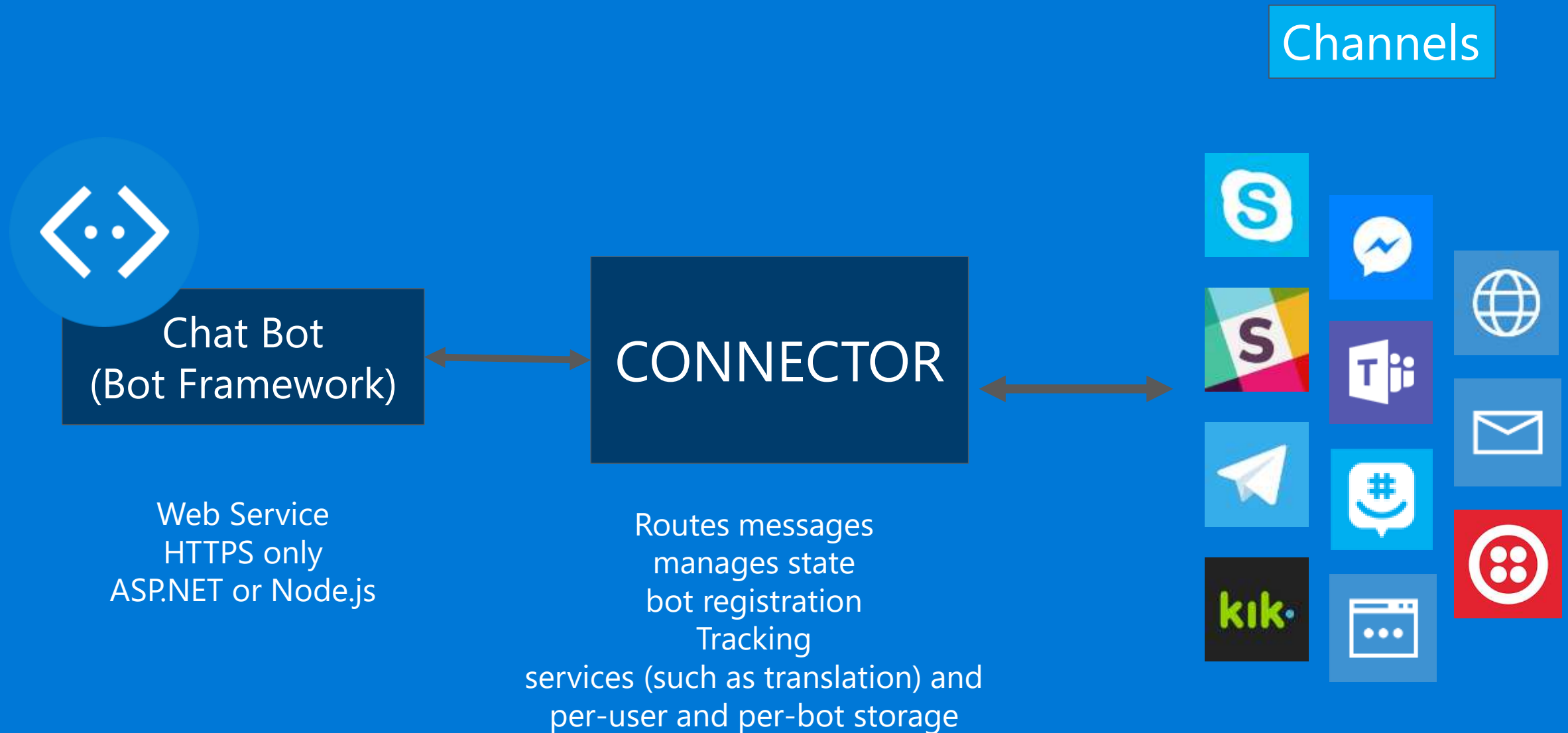
<http://docs.botframework.com>



## Options:

- Connector
- Builder
- Directory

# Connector Service Flow.



# Installing Tools

- **Visual Studio 2015 or higher**

<http://www.visualstudio.com>

- **Get the Visual Studio Bot Project Template**

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

- **Get the Bot Emulator**

<https://download.botframework.com/botconnector/tools/emulator/publish.htm>

# 자세한 내용은 Github

<http://github.com/KoreaEva/Bot>





# DEMO

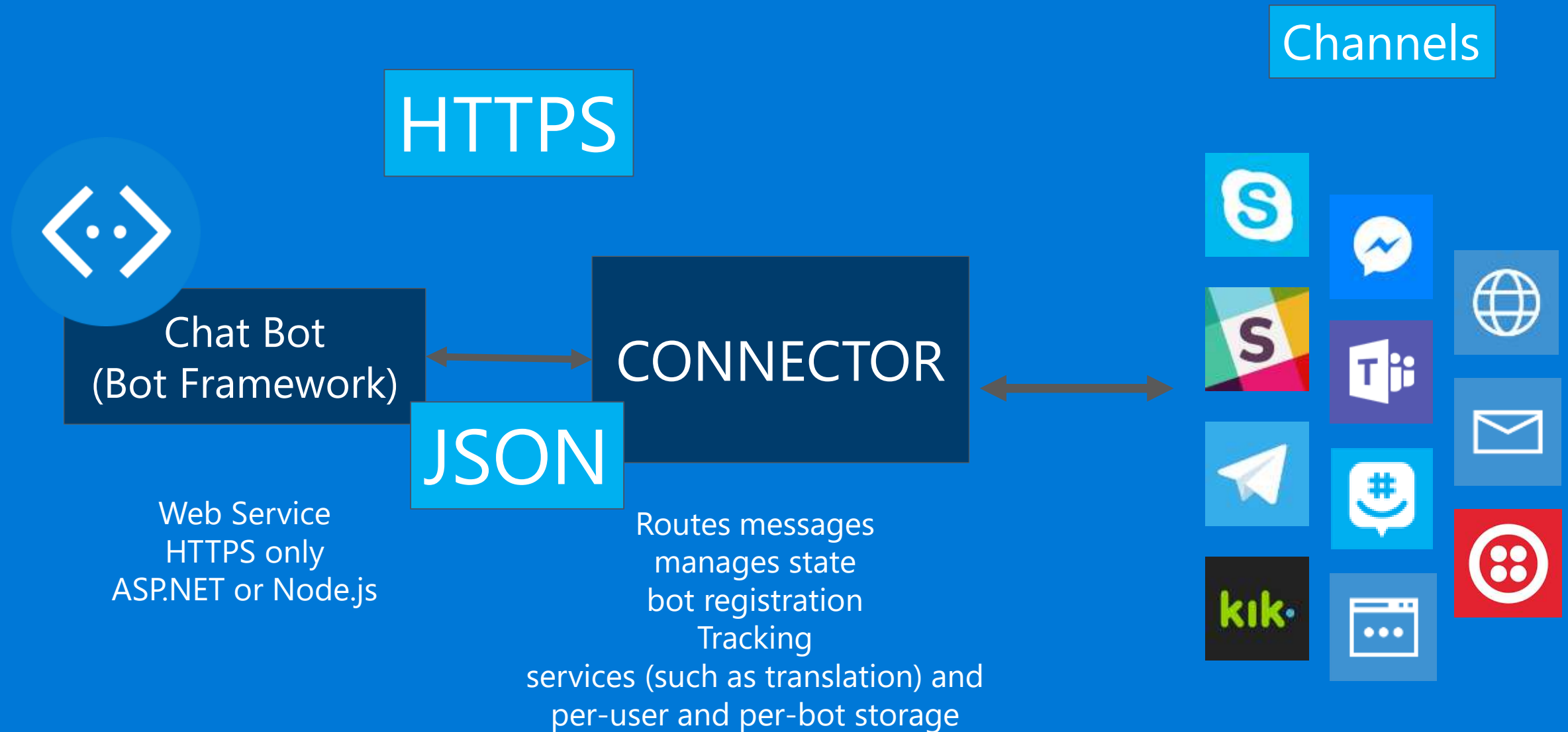
Hello Bot



# Connector, Activities & Messages

Connector	The Connector (or Connector Service) handles all communication, conversations, state, and authorization for all activities between a Bot and Users.
Activity	An Activity is a specific event that occurs between a Bot and Users, such as an actual message, or conversation notification.
Message	A Message is an overt (typically visible) communication between a Bot and Users, such as a User asking a question, or a Bot responding with a reply.

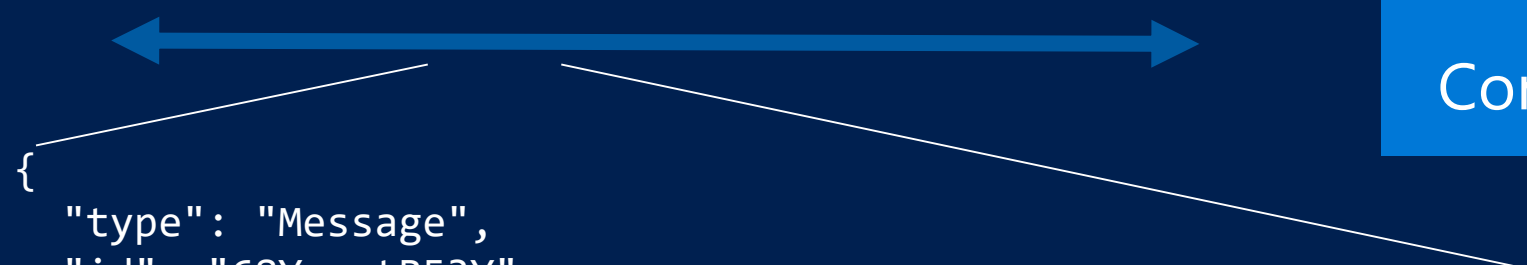
# Connector Service Flow.



# Bot Connector messages

Your bot

Bot  
Connector



```
{
  "type": "Message",
  "id": "68YrxgtB53Y",
  "conversationId": "DphPaFQrDuZDKyCez4AFGcT4vy5aQDje1lLGIjB8v18MFtb",
  "language": "en",
  "text": "You can say \"/order\" to order!",
  "attachments": [ ],
  "from": {
    "name": "+12065551212",
    "channelId": "sms",
    "address": "+12065551212",
    "id": "Ro52hKN287",
    "isBot": false
  },
  "channelData": { SMS data here },
  "botUserData": { your data here },
  ...
}
```



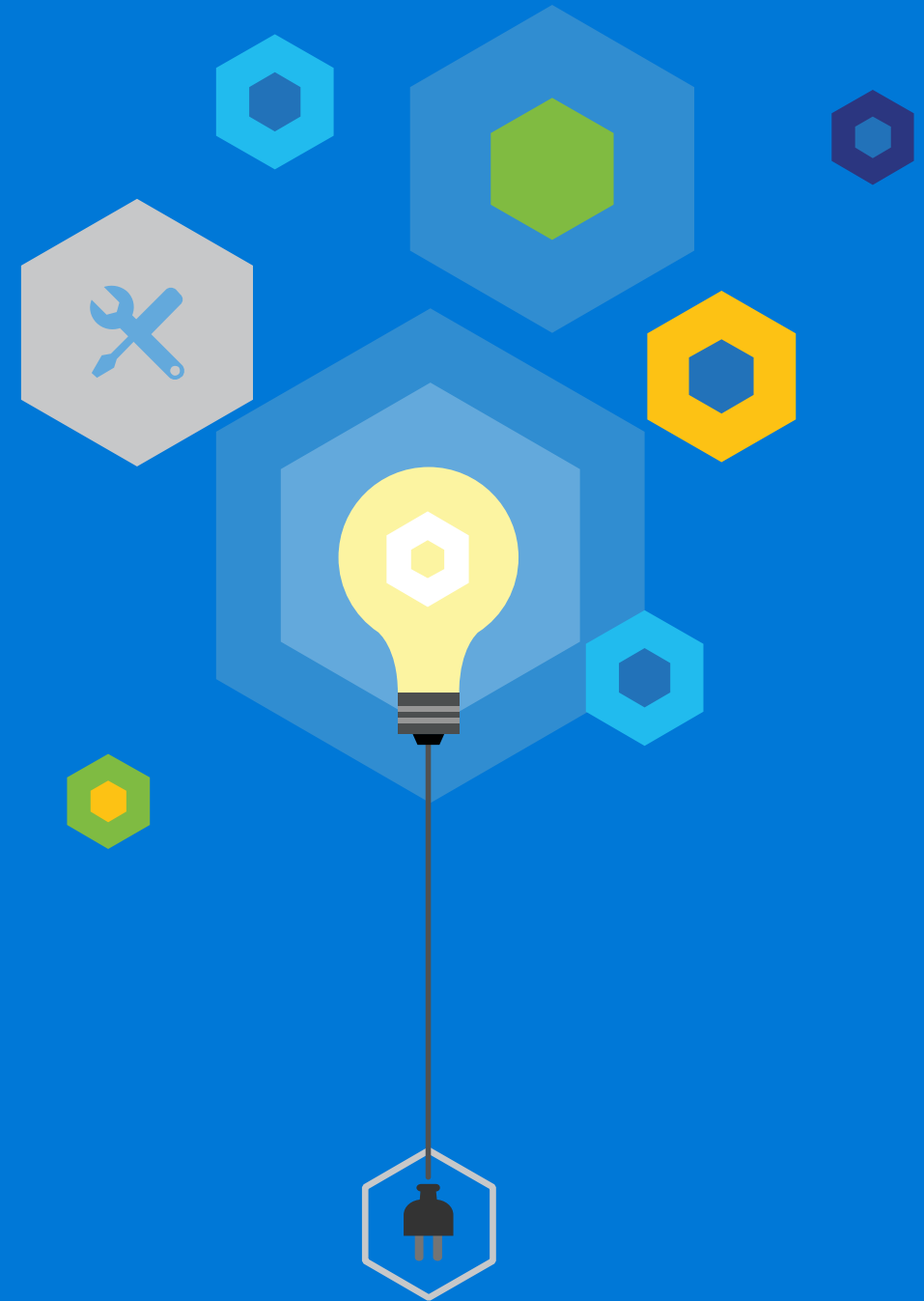
# Connector

Namespace: Microsoft.Bot.Connector

```
ConnectorClient connector = new ConnectorClient(  
    new Uri(activity.ServiceUrl));  
  
string message = string.Format("{0}을 주문 받았습니다. 감사합니다.",  
    activity.Text);  
  
// return our reply to the user  
Activity reply = activity.CreateReply(message);  
await connector.Conversations.ReplyToActivityAsync(reply);
```

# DEMO

## Connector Activities & Messages



# Types of Activities

Activity Type	Description
Message	Sent when general content is passed to or from a user and a bot
Conversation Update	Sent when the conversation's properties change, for example the topic name, or when user joins or leaves the group
Contact Relation Update	Sent when bot added or removed to contact list
Delete User Data	Send when user is removed from a conversation
Typing	Sent when a user is typing
Ping	Send when a keep-alive is needed

# Activities Types

```
switch (activity.GetActivityType())
{
    case ActivityTypes.Message:

        message = string.Format("{0}을 주문 받았습니다. 감사합니
        reply = activity.CreateReply(message);
        await connector.Conversations.ReplyToActivityAsync(r

        break;

    case ActivityTypes.ConversationUpdate:
        message = string.Format("안녕하세요 만리장성 봇 입니다.

        reply = activity.CreateReply(message);
```



```
case ActivityTypes.ConversationUpdate:
    message = string.Format("안녕하세요 만리장성 봇 입니다. 주문하실

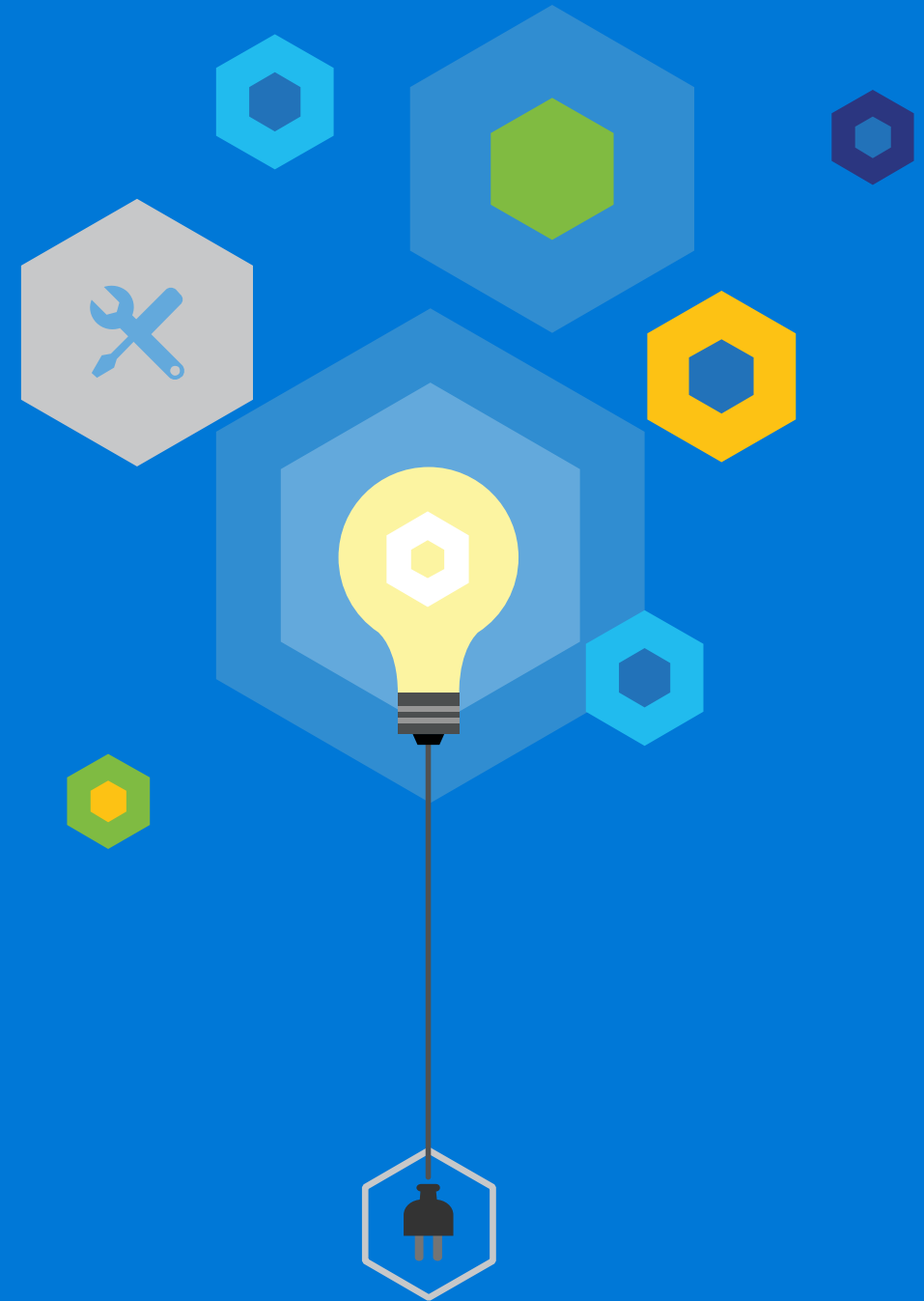
    reply = activity.CreateReply(message);
    await connector.Conversations.ReplyToActivityAsync(reply);

    break;

case ActivityTypes.ContactRelationUpdate:
case ActivityTypes.Typing:
case ActivityTypes.DeleteUserData:
default:
    break;
}
```

# DEMO

Type of Activities

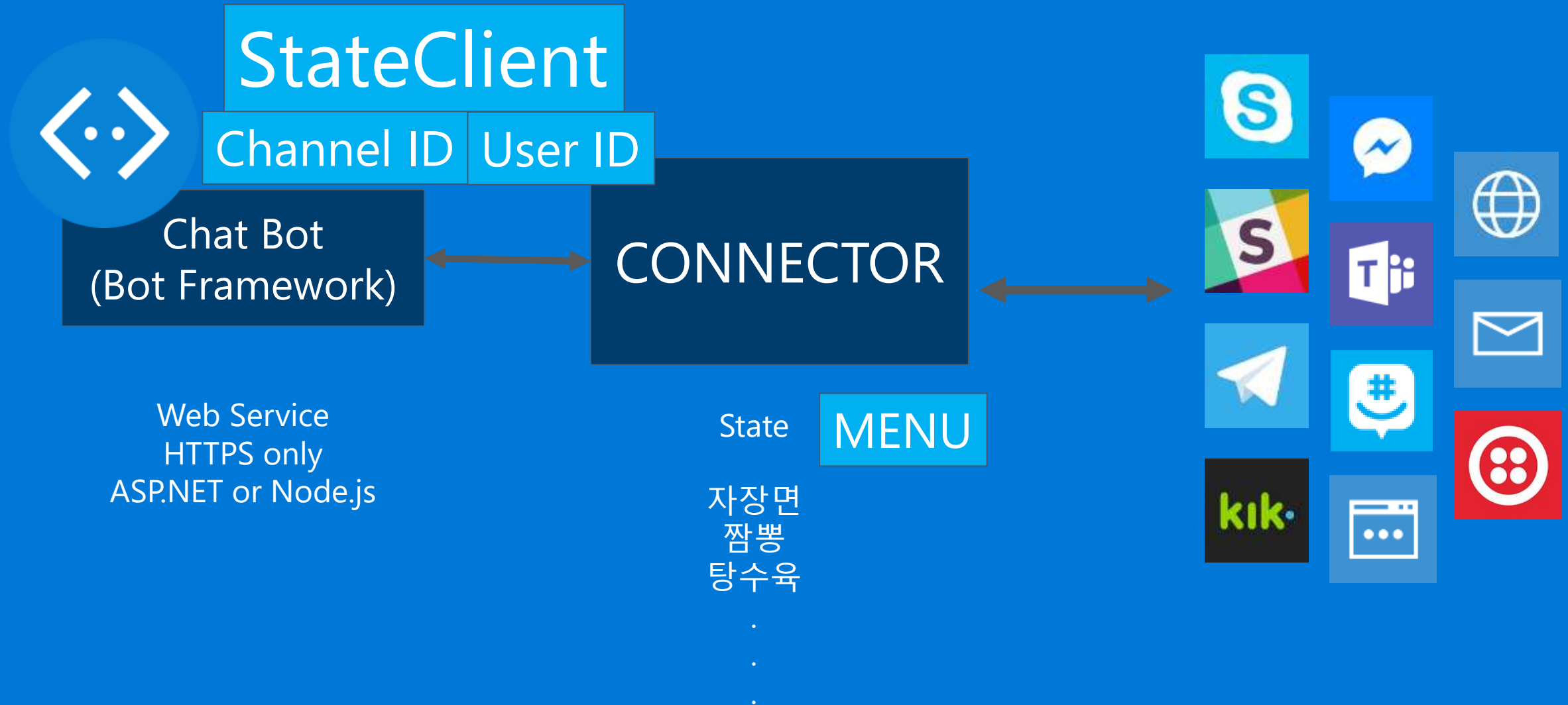


# Managing State

Not all Bots require state management outside of what the Connector provides inherently, however if you need to manage state, there are various methods.

- Custom storage methods, such as a remote database
- Class-specific logic at runtime
- Implicitly during a Dialog or Form process
- The Bot Framework State Client

# Connector Service Flow.





# StateClient

```
StateClient stateClient = activity.GetStateClient();
BotData userData = await stateClient.BotState.GetUserDataAsync(
    activity.ChannelId, activity.From.Id);

string menu = activity.Text;
menu += userData.GetProperty<string>("MENU") + ",";

userData.SetProperty<string>("MENU", menu);
await stateClient.BotState.SetUserDataAsync(
    activity.ChannelId, activity.From.Id, userData);

reply = activity.CreateReply("주문내역:" + menu);
await connector.Conversations.ReplyToActivityAsync(reply);
```

# StateClient

```
case ActivityTypes.DeleteUserData:  
    await stateClient.BotState.DeleteStateForUserAsync  
        (activity.ChannelId, activity.From.Id);  
  
break;
```

# DEMO

State Client



# Using Dialogs

Dialogs are a way of wrapping an entire “experience” into an easily managed interaction based on a “chained” and “conversational” paradigm.

- Send information to a User
- Prompt a User for more information or confirmation
- Provide conditional logic
- Provide “as you need it” content
- Dialogs can contain or forward to other Dialogs.

# Connector Service Flow.



# Dialog

Namespace: Microsoft.Bot.Builder.Dialogs

```
[Serializable]
public class EchoDialog : IDialog<object>
{
    public async Task StartAsync(IDialogContext context)
    {
        context.Wait(MessageReceivedAsync);
    }
    public async Task MessageReceivedAsync(IDialogContext context,
    {
        context.Wait(MessageReceivedAsync);
    }
}
```

# Dialog

Namespace: Microsoft.Bot.Builder.Dialogs

```
[Serializable]
public class EchoDialog : IDialog<object>
{
    public async Task StartAsync(IDialogContext context)
    {
        context.Wait(MessageReceivedAsync);
    }
    public async Task MessageReceivedAsync(IDialogContext context,
    {
        context.Wait(MessageReceivedAsync);
    }
}
```



# DEMO

Using State, Authentication &  
Debugs



# Using Forms with FormFlow

Although Dialogs are the basic building block of a conversation, it's difficult to create a “guided” conversation. FormFlow creates Dialogs and guides a User through filling in a “form” while providing help and guidance along the way.

- Super smart based on enumerations and properties
- Provide content for various scenarios, such as not understanding, clarification, and confirmations
- Easily provide optional conversation paths based on previous choices

# Connector Service Flow.



# FormFlow

Namespace: Microsoft.Bot.Builder.FormFlow

```
[Serializable]
public class FoodOrder
{
    public FooldOptions? Food;
    public LengthOptions? Length;

    public static IForm<FoodOrder> BuildForm()
    {
        return new FormBuilder<FoodOrder>()
            .Message("만리장성에 오신 여러분을 환영합니다.")
            .Build();
    }
}
```

# FormFlow

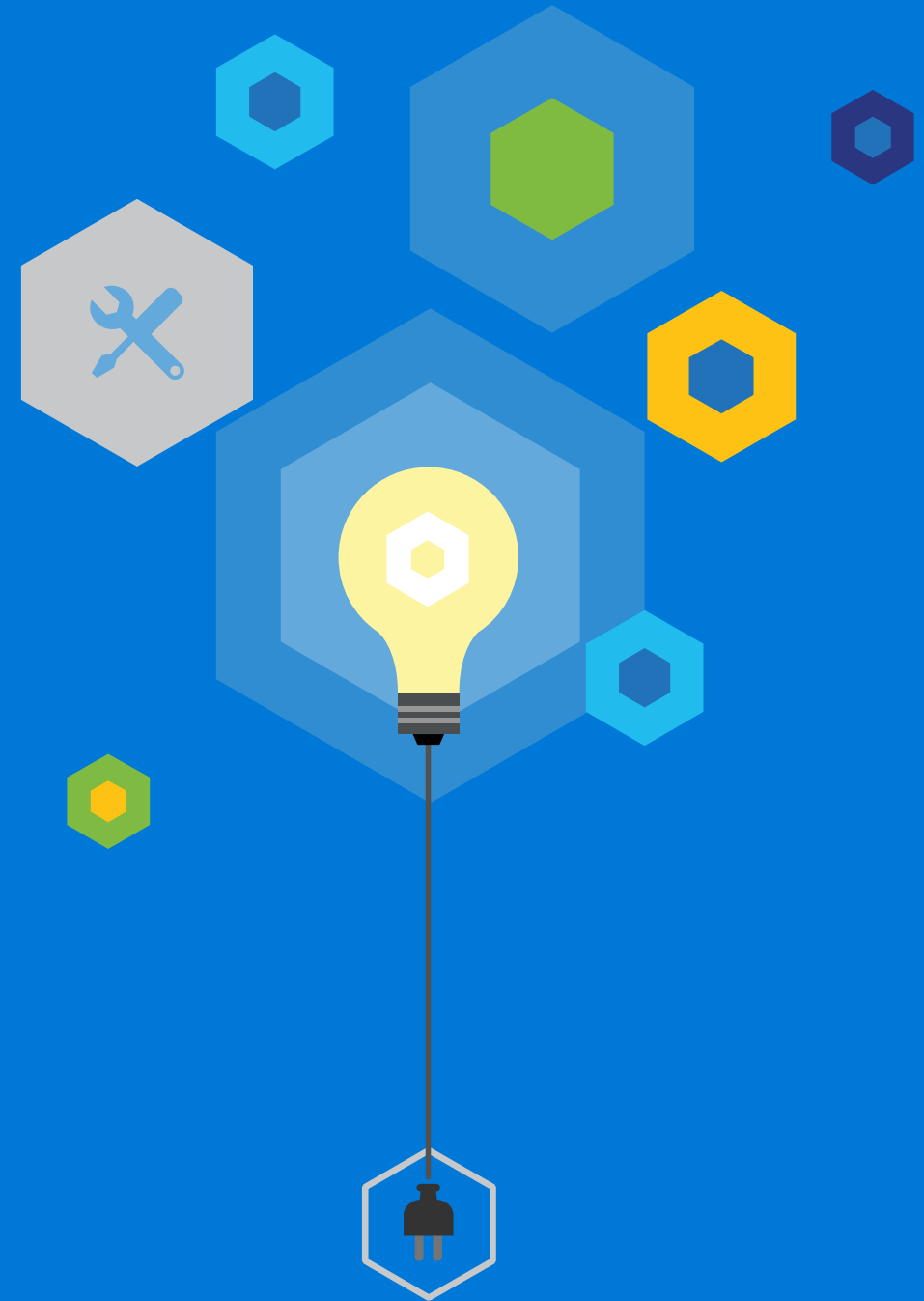
Namespace: Microsoft.Bot.Builder.FormFlow

```
public enum FooldOptions
{
    자장면, 짬뽕, 탕수육, 기스면, 란자완스
};
```

```
public enum LengthOptions { 보통, 곱배기};
```

# DEMO

Using Forms with FormFlow  
and Bot Deployment



# Azure Function

The screenshot displays the Azure Functions portal interface for a function named 'myBlob (Azure Storage Blob)'. The top navigation bar includes 'Develop', 'Integrate', and 'Monitor' tabs. Below this, a diagram shows the function's architecture: a 'Trigger' (lightning bolt icon) points to an 'Input' (cylinder icon), which points to an 'Output' (cylinder icon). The 'Trigger' is configured with 'myBlob (Azure Storage Blob)'. The 'Input' and 'Output' sections are currently empty, with a '+ New input' button available.

The 'Code' tab is selected, showing the following C# code:

```
1 #r "Microsoft.WindowsAzure.Storage"
2 #r "System.Runtime"
3 #r "System.Threading.Tasks"
4 #r "System.IO"
5
6 using System;
7 using System.Threading.Tasks;
8 using Microsoft.WindowsAzure.Storage.Blob;
9 using Microsoft.ProjectOxford.Vision;
10
11 public static async Task Run(ICloudBlob myBlob, TraceWriter log, IAsyncCollector<object> document)
12 {
13     var visionClient = new VisionServiceClient("e6dcf6fa3e4942ac81042bfd1d8af235");
14     var result = await visionClient.RecognizeTextAsync(myBlob.Uri.ToString(), "en");
15
16     var words = from r in result.Regions
17                 from l in r.Lines
18                 from w in l.Words
19                 select w.Text;
20 }
```

The 'Logs' tab is also visible, showing a series of log entries from the function's execution. The logs indicate that the function was successfully triggered and completed, with the recognized text from the blob being logged.

myBlob (Azure Storage Blob) delete

Name the blob for use in your code ⓘ

myBlob

What path should the trigger monitor? ⓘ

ocme

Behavior ⓘ

trigger

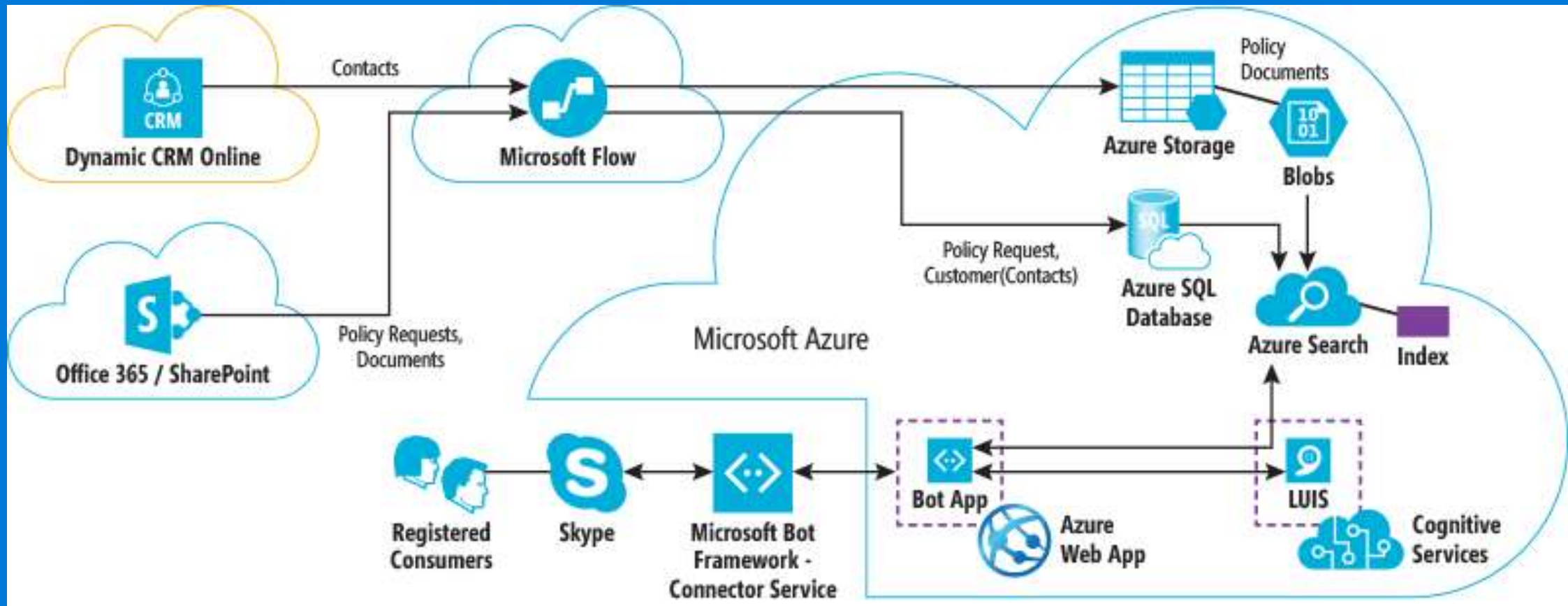
Which stora

Logs ⓘ Pause ⓘ Clear

2016-03-29T16:53:18 Welcome, you are now connected to log-streaming service.  
2016-03-29T16:54:18 No new trace in the past 1 min(s).  
2016-03-29T16:54:23.661 Function started (Id=821a788a-7c72-434a-87f5-e70738fb1fc5)  
2016-03-29T16:54:23.786 Function started (Id=8423dc01-533d-43d8-a61e-3babe1d85660)  
2016-03-29T16:54:23.879 Function started (Id=f2262b30-b1e6-4c87-8be3-a735898ab602)  
2016-03-29T16:54:24.696 Recognized words: Ella Mashkouski  
2016-03-29T16:54:24.712 Recognized words: Menu p Home Current Statement Posted Transactions DATE v MAR 3 Doing business as: Statements & Activity AMERICAN  
2016-03-29T16:54:24.884 Function completed (Success, Id=821a788a-7c72-434a-87f5-e70738fb1fc5)  
2016-03-29T16:54:25.182 Function completed (Success, Id=8423dc01-533d-43d8-a61e-3babe1d85660)  
2016-03-29T16:54:27.032 Recognized words: Microsoft Completion eLesson: Unconscious Bias Closing Remarks/Completion 1 Completion Contents CREDIT RECEIVED  
2016-03-29T16:54:27.216 Function completed (Success, Id=f2262b30-b1e6-4c87-8be3-a735898ab602)



# Azure Search



# GitHub!

The screenshot shows the GitHub web interface for the Microsoft/BotBuilder repository. The browser's address bar displays 'github.com/Microsoft/botbuilder'. The repository page header includes navigation links like 'Personal', 'Open source', 'Business', and 'Explore', along with a search bar and 'Sign in'/'Sign up' buttons. The repository name 'Microsoft / BotBuilder' is prominently displayed, accompanied by statistics: 198 watchers, 2,131 stars, and 374 forks. Below this, tabs for 'Code', 'Issues' (72), 'Pull requests' (1), 'Pulse', and 'Graphs' are visible. The main content area contains a descriptive paragraph about the Microsoft Bot Builder SDK, its role in the Microsoft Bot Framework, and a link to 'http://botframework.com'. A progress bar indicates repository activity with 706 commits, 4 branches, 29 releases, and 26 contributors. At the bottom, there are buttons for 'New pull request', 'Find file', and 'Clone or download'. A recent commit by 'Stevenic' is listed, showing a merge of pull request #409 and the latest commit hash 'c2dcafc' from 9 days ago. A file named 'CSharp' is shown with an update to version 1.2.3.0 from 12 days ago.

GitHub - Microsoft/BotBuilder

Personal Open source Business Explore Pricing Blog Support This repository Search Sign in Sign up

Microsoft / BotBuilder

Watch 198 Star 2,131 Fork 374

Code Issues 72 Pull requests 1 Pulse Graphs

The Microsoft Bot Builder SDK is one of three main components of the Microsoft Bot Framework. The Microsoft Bot Framework provides just what you need to build and connect intelligent bots that interact naturally wherever your users are talking, from text/SMS to Skype, Slack, Office 365 mail and other popular services. <http://botframework.com>

706 commits 4 branches 29 releases 26 contributors

Branch: master New pull request Find file Clone or download

Stevenic Merge pull request #409 from mulyoved/master Latest commit c2dcafc 9 days ago

CSharp Update version to 1.2.3.0 12 days ago

<https://github.com/Microsoft/botbuilder>

감사합니다

Thank you~!

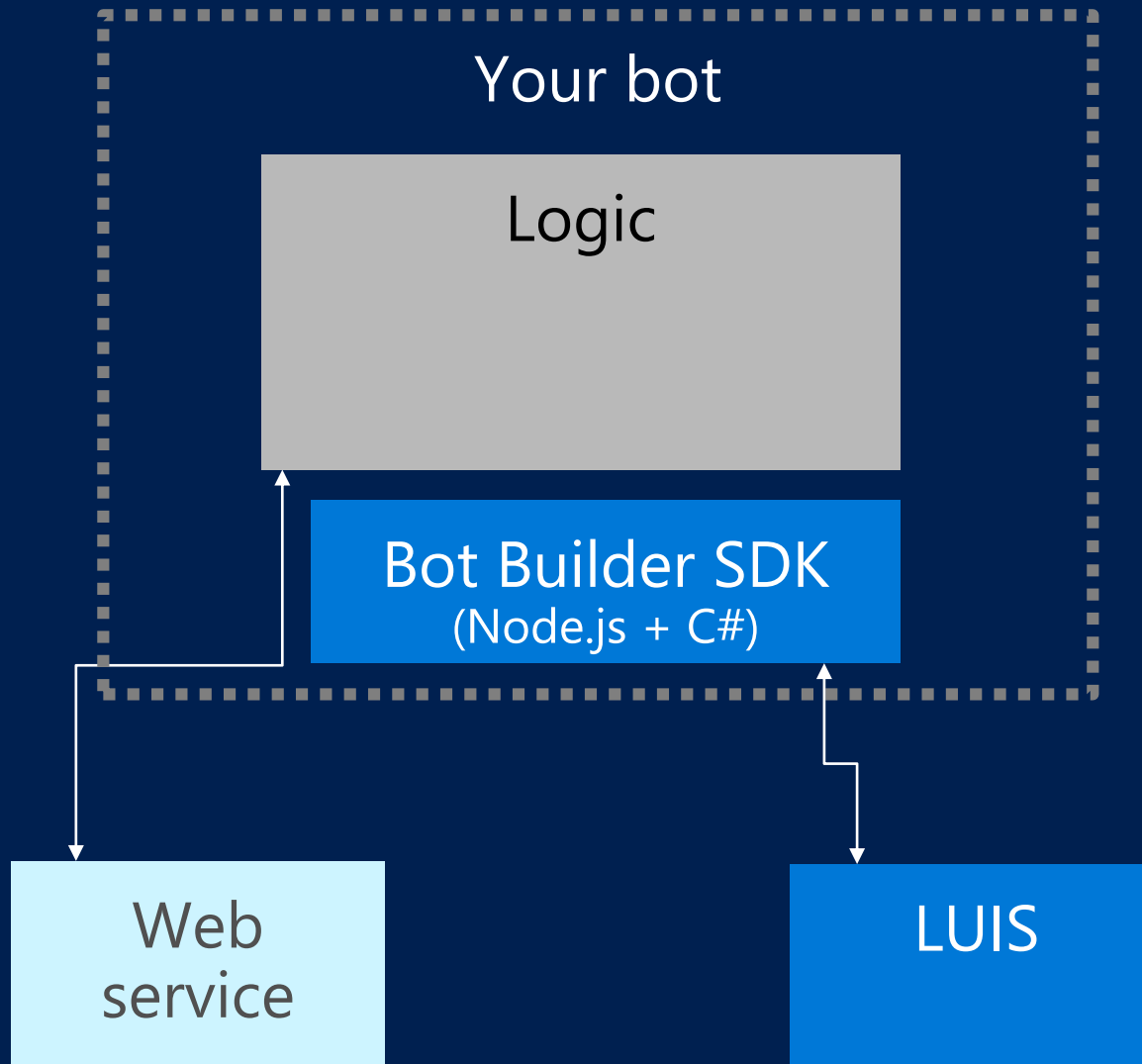
# Tips & Tricks

- Use an empty string for your Microsoft App Id and Password in development, but remember to change these to your actual values before you publishing into production.
- Get used to understanding Lambda and “Fluid” concepts to make it easier to work with Dialogs, Forms and especially the concept of the Bot process “Chain”.
- This session demonstrates storing and managing Bot State is a number of ways. Use what makes sense in your environment by uses these concepts purely as examples.
- If the Microsoft Bot Emulator continues to give you authorization issues, try changing the port for your Bot Service in the Web tab of the project and redeploy.

# Integrating Language Understanding Intelligence Services

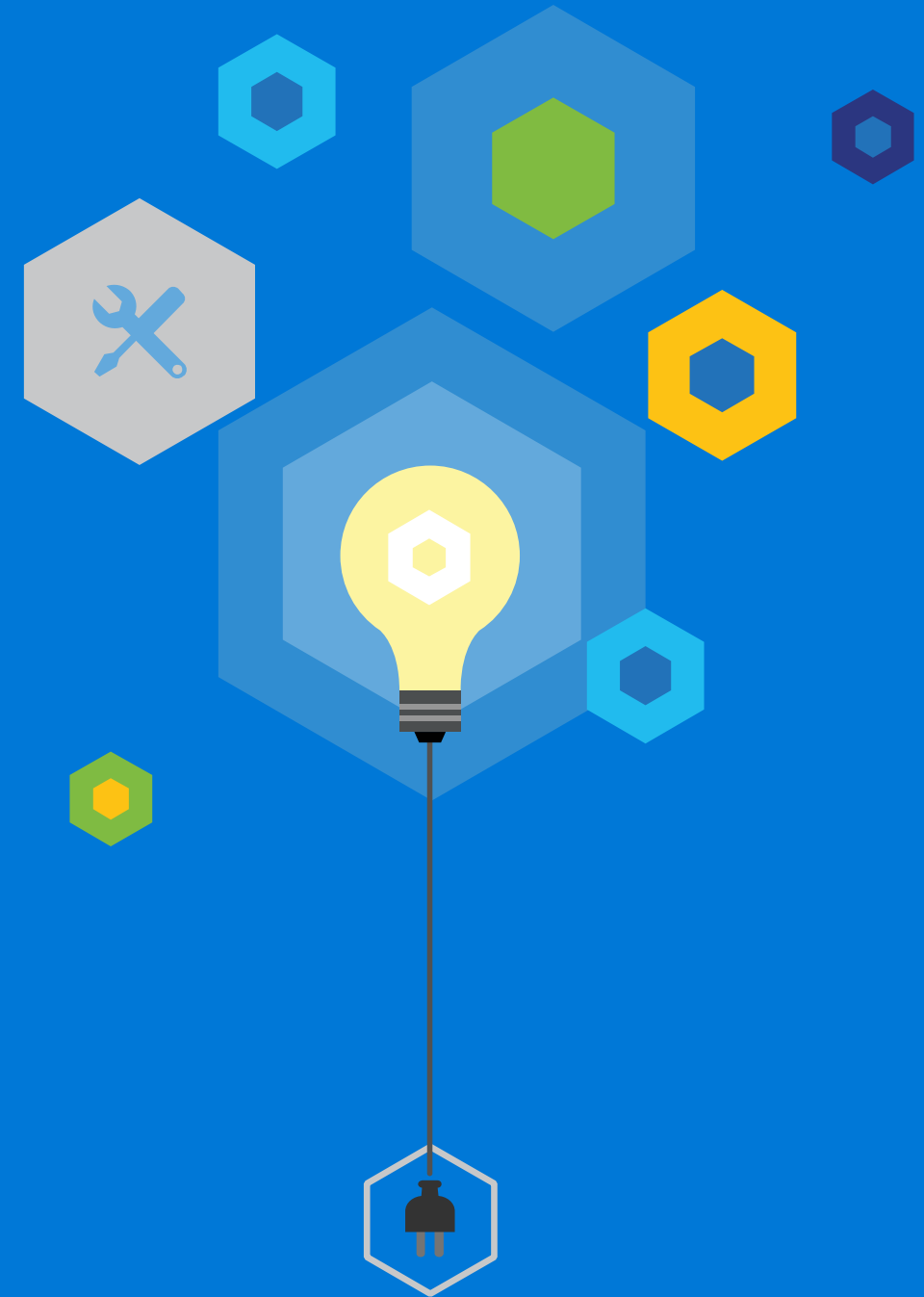
LUIS is part of Microsoft Cognitive Services offering and can be used for any device, on any platform, and any application scenario.

- Essentially what Cortana uses on the backend for language and semantic interpretations
- Provides “built in” logic that can be leveraged “out of the box”
- Natural, adaptable, conversational intelligence
- Model-based via intents and entities



# DEMO

Integration Language Understanding  
Intelligence Services





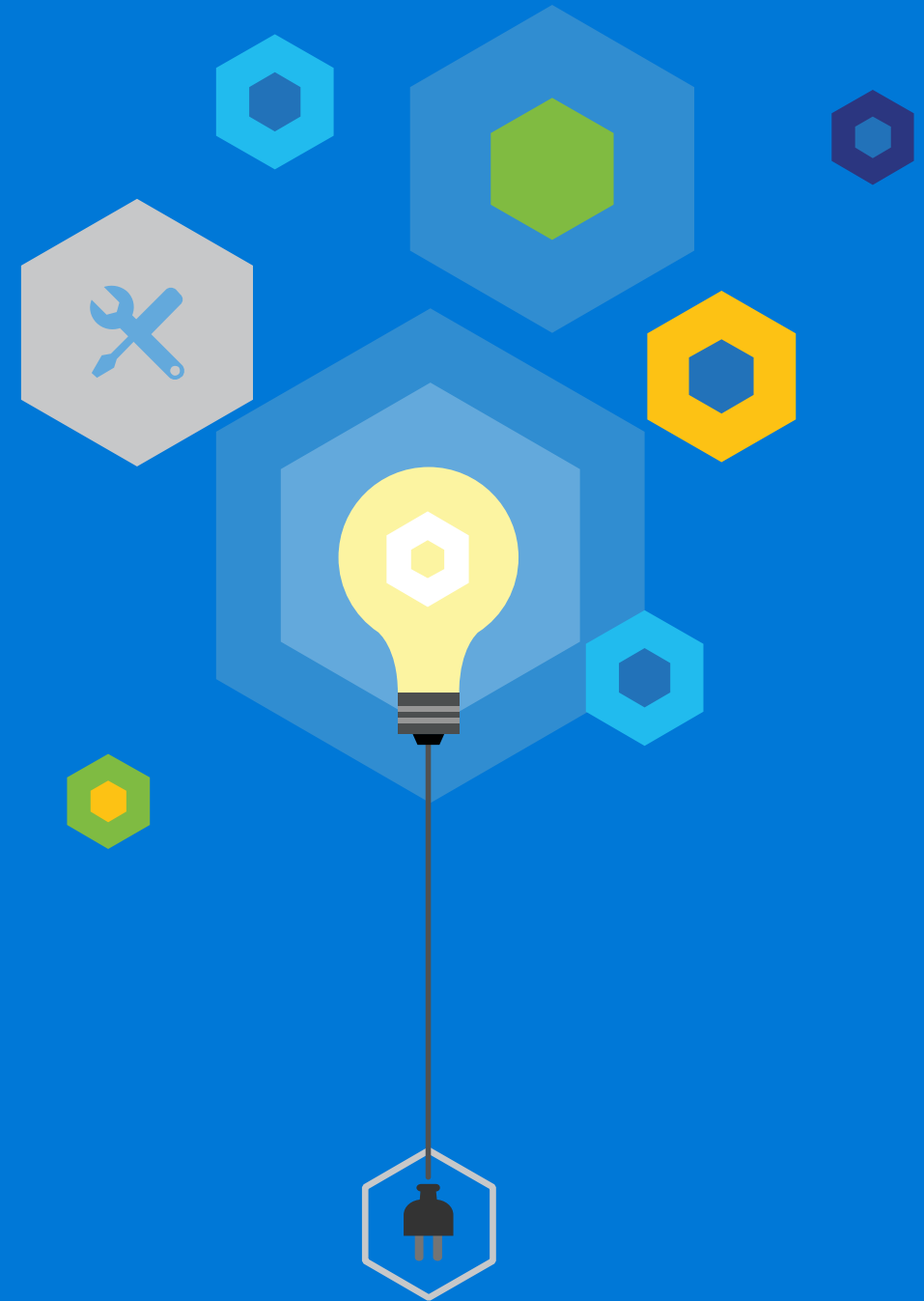
# Putting it All Together

- Add some more features to demonstrate LUIS integration
- Walkthrough an example of using Attachments, Cards and Actions
- Publishing your Bot
- Using your Bot in a web chat and Skype

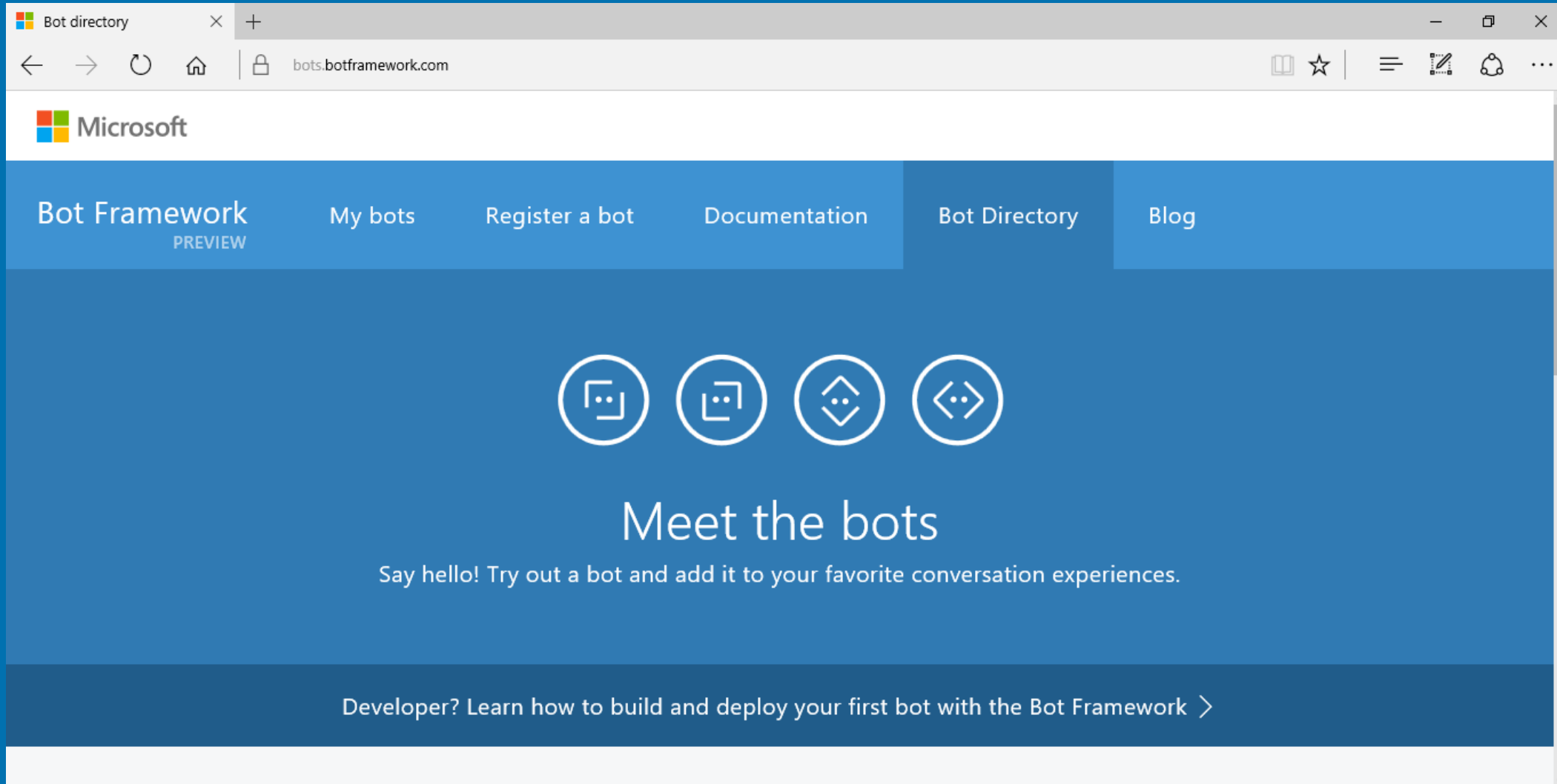
Putting it All Together

# DEMO

Putting It All Together



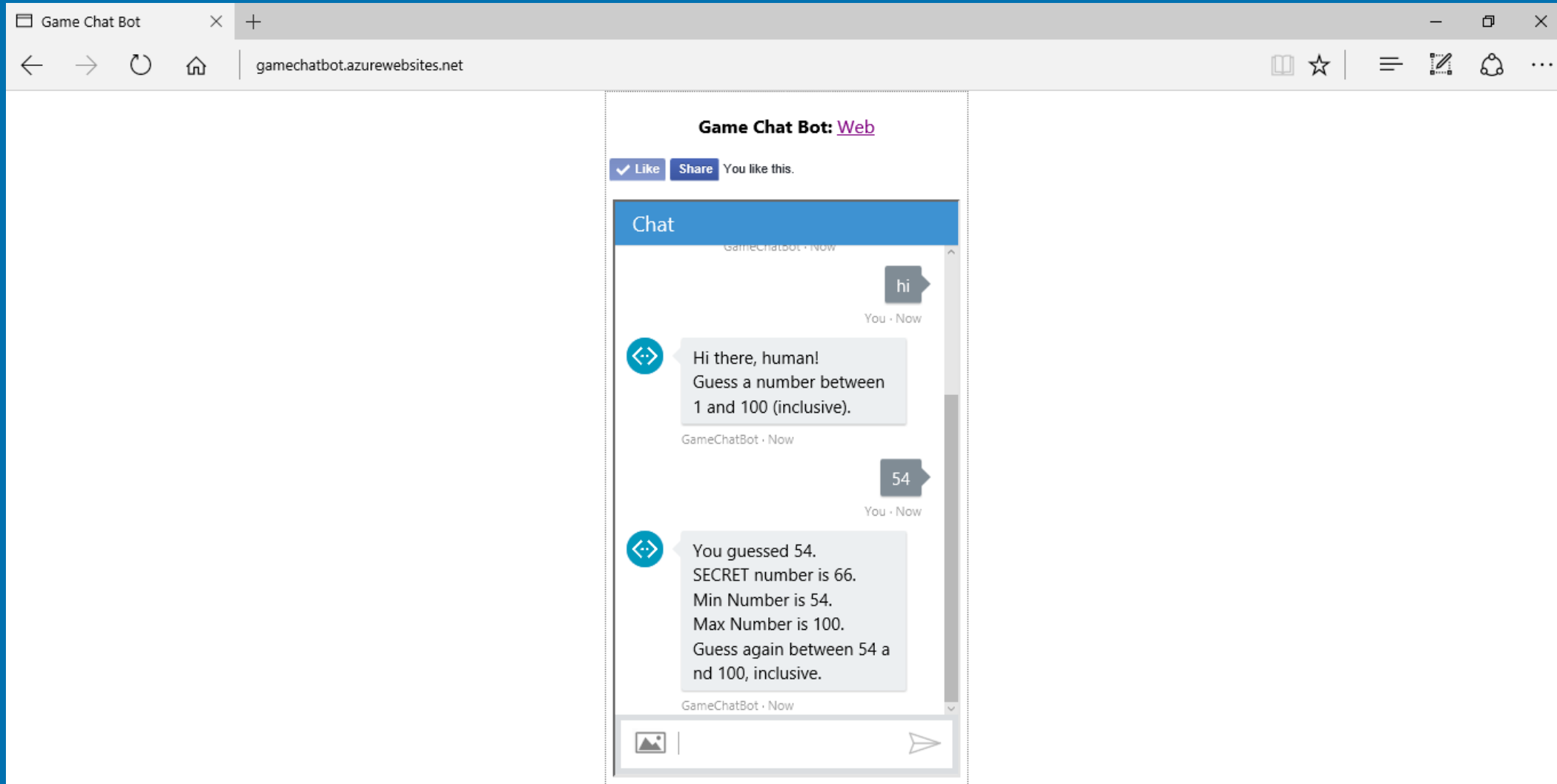
# Bot Directory



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

Demo

# Game Chat Bot



<http://gamechatbot.azurewebsites.net/>



40 OLD  
TOWN

BIRKSEN

(020)  
76226466

40 OLD  
TOWN

BIRKSEN

(020)  
76226466

flowers by



# Variety of Creative Apps

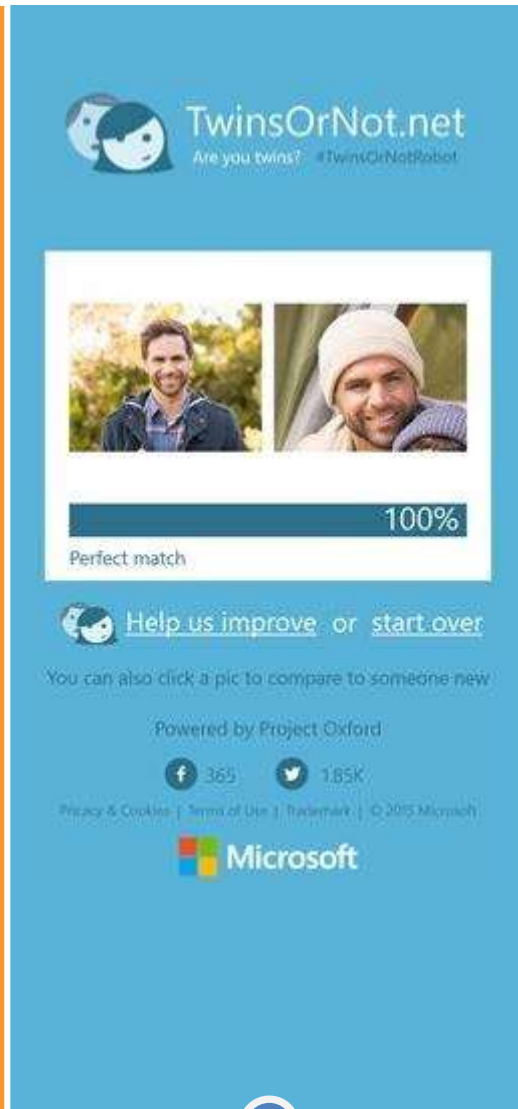


How-Old.net  
HOW OLD DO I LOOK? #HowOldNet

4

Sorry if we didn't quite get the age and gender right - we are still working on this feature

Try Another Photo!



TwinsOrNot.net  
Are you twins? #TwinsOrNotRobot

100%  
Perfect match

Help us improve or start over

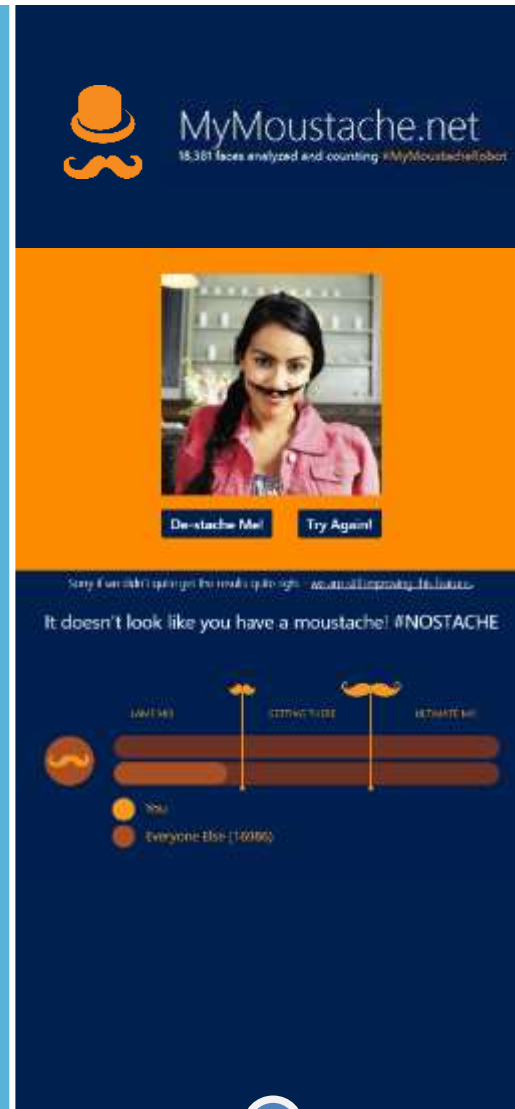
You can also click a pic to compare to someone new

Powered by Project Oxford

365 1.85K

Privacy & Cookies | Terms of Use | Trademark | © 2015 Microsoft

Microsoft



MyMoustache.net  
15,381 faces analyzed and counting #MyMoustacheRobot

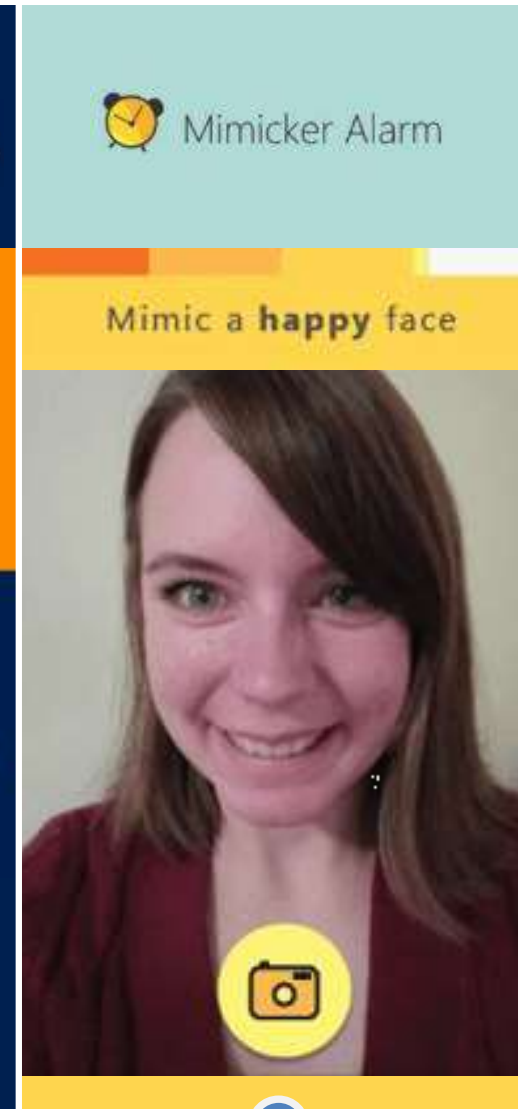
De-stache Me! Try Again!

Sorry if we didn't quite get the moustache quite right - we are still improving this feature

It doesn't look like you have a moustache! #NOSTACHE

WAVE ME! GETTING THERE! ULTIMATE ME!

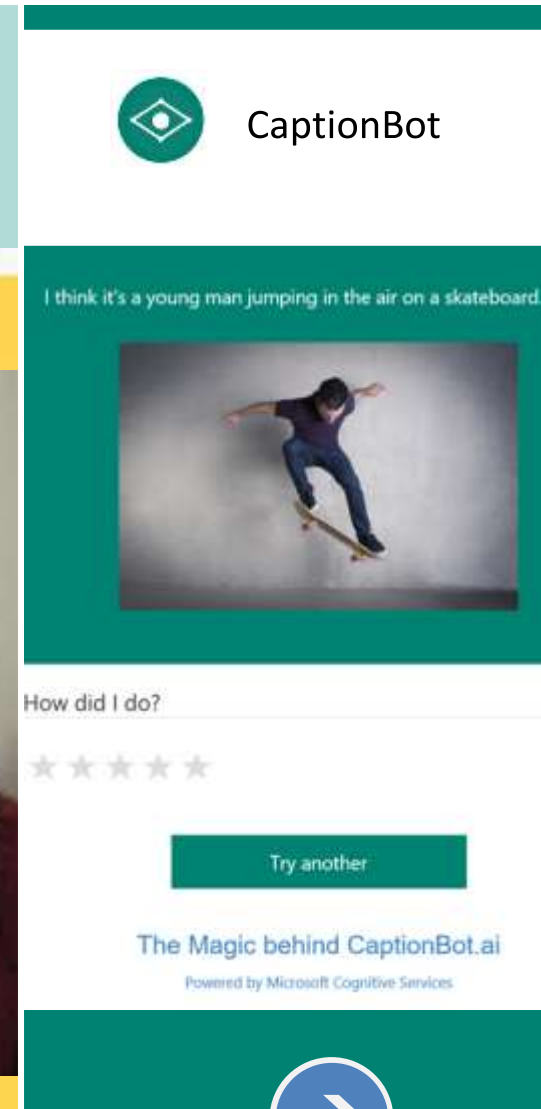
You  
Everyone Else (16086)



Mimicker Alarm

Mimic a happy face

Camera icon



CaptionBot

I think it's a young man jumping in the air on a skateboard

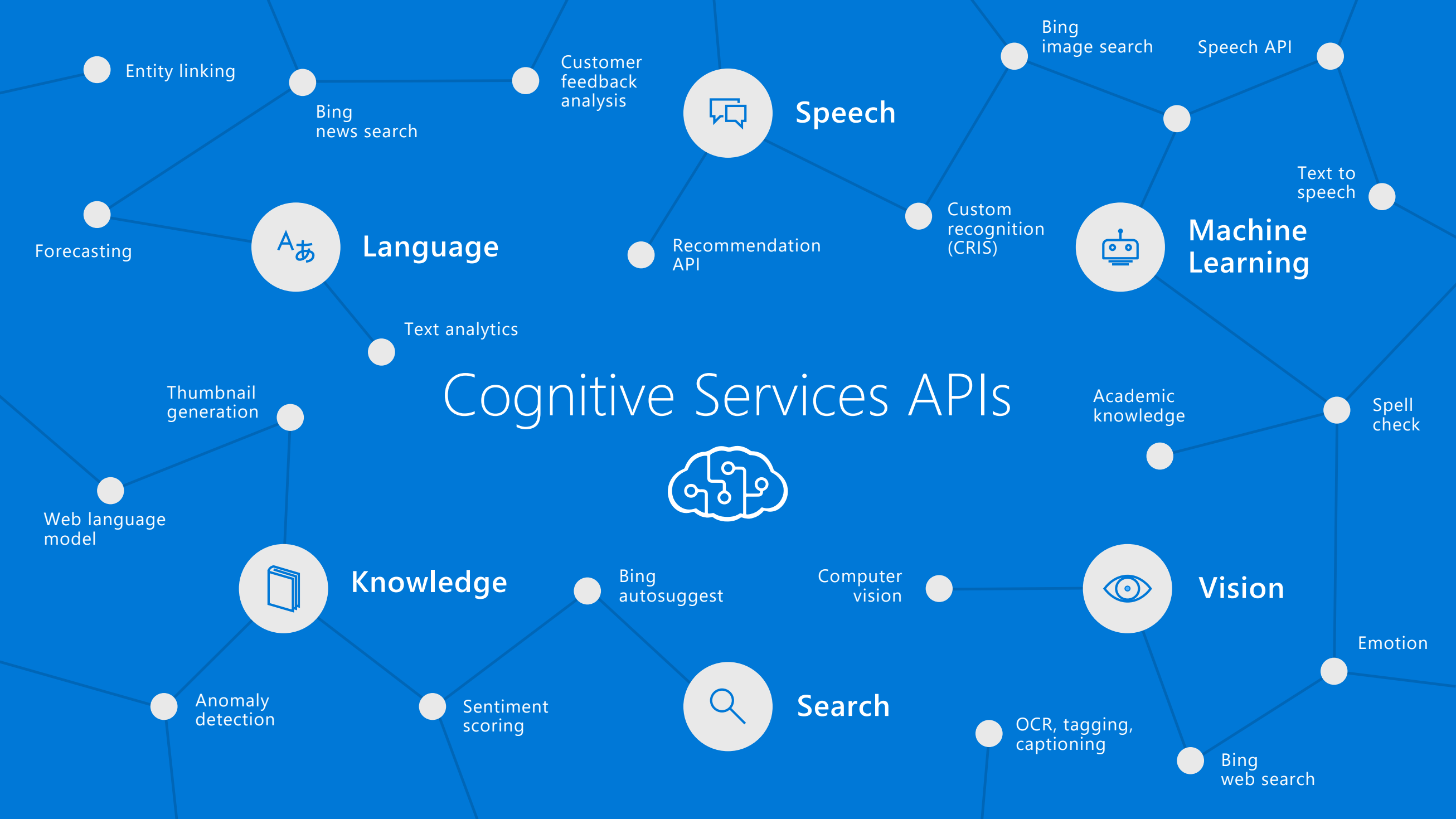
How did I do?

★★★★★

Try another

The Magic behind CaptionBot.ai  
Powered by Microsoft Cognitive Services






Next arrow icon





# Cognitive Services

microsoft.com/cognitive

 Vision	 Speech	 Language	 Knowledge	 Search
Computer Vision	Custom Recognition	Bing Spell Check	Academic Knowledge	Bing Web Search
Emotion	Speaker Recognition	Linguistic Analysis	Entity Linking	Bing Image Search
Face	Speech	Language Understanding	Knowledge Exploration	Bing Video Search
Video	Translator	Text Analytics	Recommendations	Bing News Search
		WebLM		Bing Autosuggest

<https://www.microsoft.com/cognitive-services/>