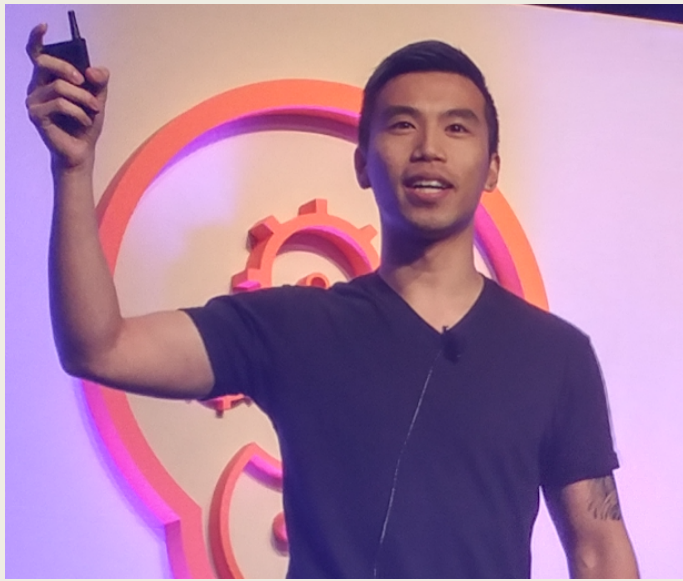


# BREAKING INTO BOTS

Introduction to Microsoft Bot Framework & LUIS.ai





# Kevin Leung

Microsoft Technical Evangelist @KSLHacks

- ✓ Microsoft Bot Framework
- ✓ Cognitive Services + LUIS.ai
- ✓ ASP.NET Core
- ✓ Internet of Things

# Gabrielle Crevecoeur

Microsoft Technical Evangelist @NoWaySheCodes

- ✓ Node.js + NodeBots
- ✓ Microsoft Bot Framework
- ✓ Azure Logic Apps
- ✓ Cognitive Services





# Road Map

- ✓ Why Bots – What is CaaP?
- ✓ Microsoft Bot Framework
- ✓ Building our own bot
- ✓ NLP Masters
- ✓ Adding Intelligence
- ✓ Resources

# Why Conversation as a Platform?

“Microsoft CEO Satya Nadella says **future belongs to bots**”

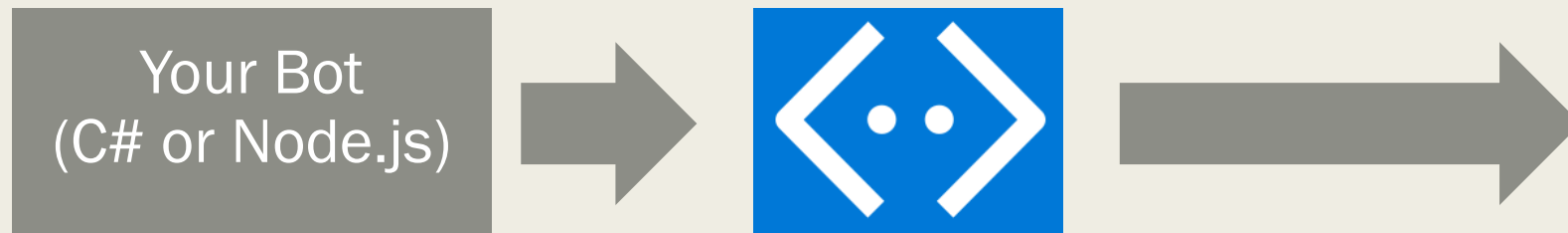










“**Bots are the new apps**,” said Nadella during a nearly three-hour keynote here that sketched a vision for the way humans will interact with machines. “People-to-people conversations, people-to-digital assistants, **people-to-bots** and even digital assistants-to-bots. **That’s the world you’re going to get to see in the years to come.**”

- USA TODAY

# Connectors and Benefits

- Connector to Platform Channels
  - *One Bot : Many Channels and endpoints*
- Why restrict yourself to one platform?
  - *Hit them all*
- Half the battle is getting users to download your app.
  - *They already have these installed*



Channels	
	Email
	GroupMe
	Skype
	Slack
	SMS
	Telegram
	Web (chat control)
	additional channels

# Microsoft Bot Framework

- Supports NodeJS and C# .NET
- Lead users through conversation
  - *Prompts, Choices, Media, Rich Cards (Displays)*
- Bot Emulator provided for local development/testing
- Easy to incorporate Natural Language Processing (Cognitive Services)
- Easily deploy to Azure Cloud Services



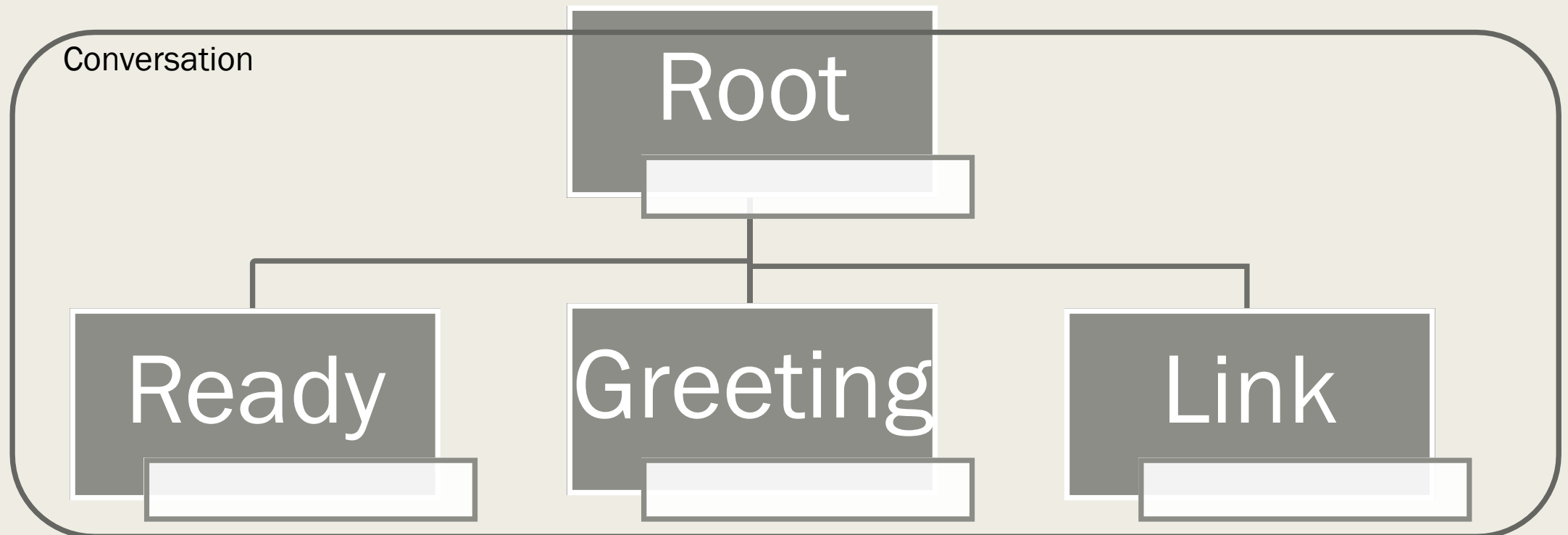
# DIALOGS, PROMPTS, INPUT & RICH CARDS

Introduction to the MSFT Bot Framework



# Dialogs

- Take part in a specific portion of a conversation with the intent of **resolving a problem**.
  - *Conversations have multiple dialogs*



# Dialogs:

Root

Ready

Greeting

Link

```
bot.dialog('/', ...  
));
```

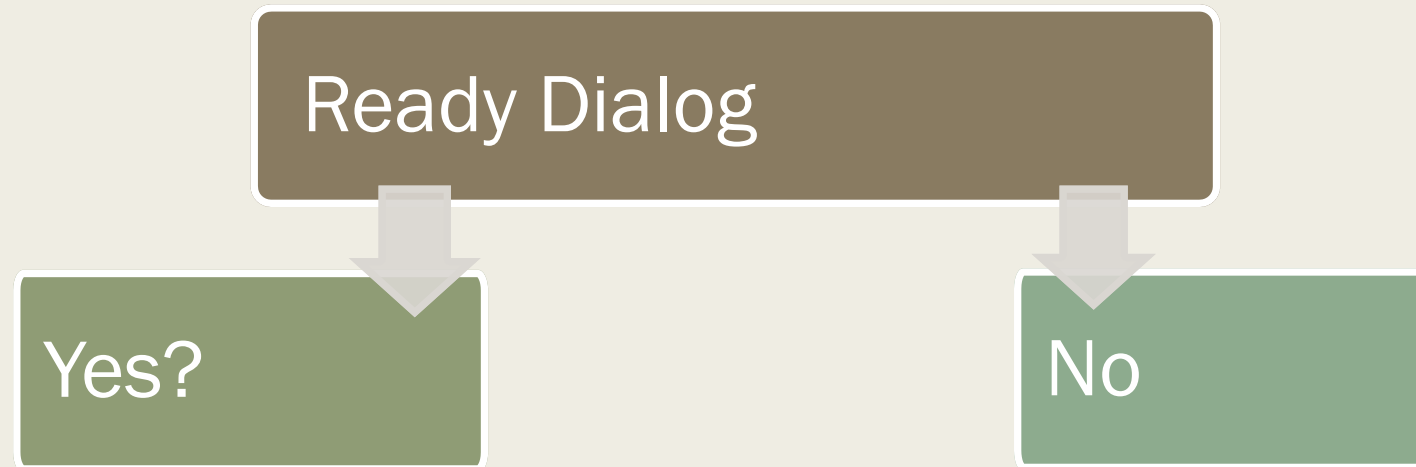
```
bot.dialog('/Ready', new builder.IntentDialog()  
));
```

```
bot.dialog( '/Greeting' , [  
    function (session) { ...  
    },  
    function (session, results) { ...  
    }]  
);
```

```
bot.dialog( '/Link' , [  
    function (session) { ...  
    }]  
);
```

# Intents

- Using a function to figure out what is wanted by the user



Ready Dialog



```
graph TD; A[Ready Dialog] --> B[Yes]; A --> C[No];
```

A flowchart with a brown box labeled 'Ready Dialog' at the top. Two arrows point down from this box to two green boxes below it. The left green box is labeled 'Yes' and the right green box is labeled 'No'.

Yes

No

```
bot.dialog('/Ready', new builder.IntentDialog()  
    .matches(/^yes/i, [  
        function (session) {  
            session.send('Great lets get your info' );  
            session.beginDialog( '/name' )  
            session.endDialog();  
        }])  
    .matches(/^no/i, [  
        function(session) {  
            session.send( 'Ok come back when you are ready ' )  
            session.endConversation();  
        }]  
    ));
```

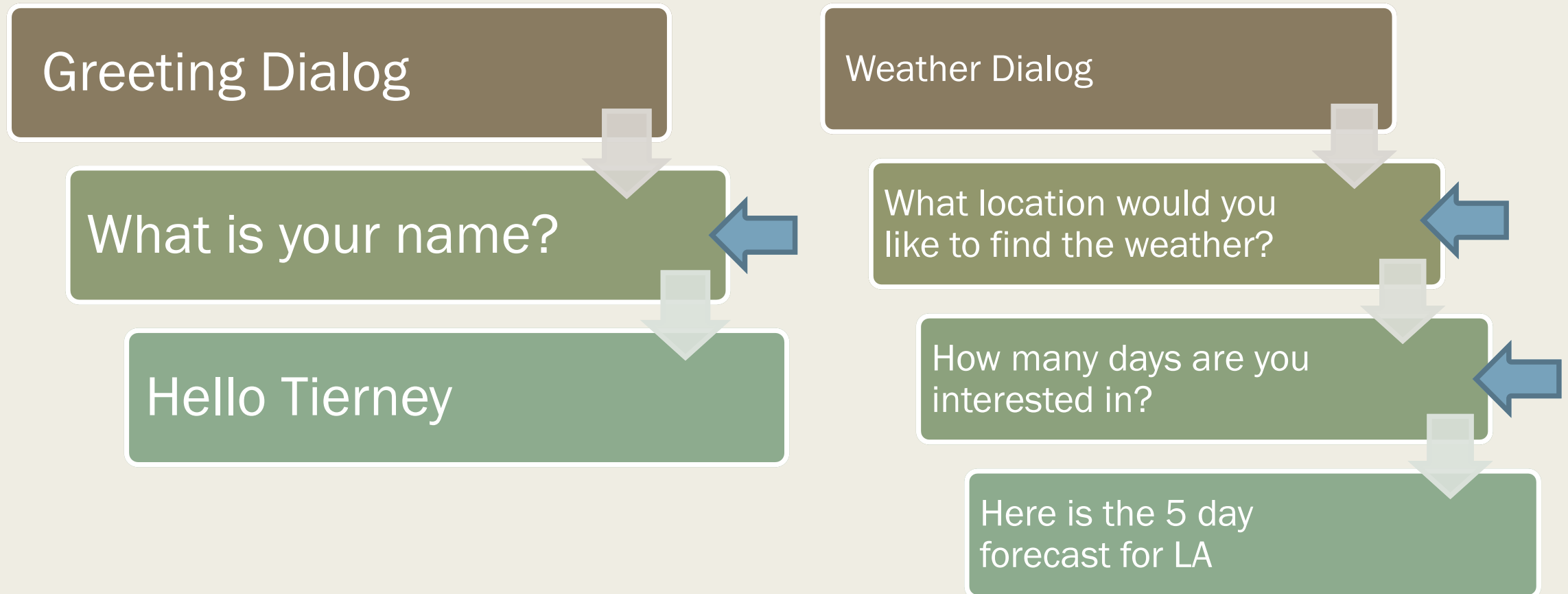
# Waterfall

- Triggering a function will create a chain of functions to execute.



# Waterfall

- Triggering a function will create a chain of functions to execute.



Greeting Dialog



```
graph TD; A[Greeting Dialog] --> B[What is your name?]; B --> C[Hello Tierney];
```

What is your  
name?

Hello Tierney

```
bot.dialog( '/Greeting' , [  
    function (session) {  
        builder.Prompts.text(session, "What is your name" )  
    },  
    function (session, results) {  
        session.send( "Hello " + results.response )  
        session.beginDialog('/Link')  
        session.endDialog();  
    }  
]);
```

## Weather Dialog

What location would you like to find the weather?

How many days are you interested in?

Here is the 5 day forecast for LA

```
bot.dialog('/Weather', [  
  function (session) {  
    builder.Prompts.text(session, 'What location?')  
  },  
  function (session, results) {  
    // results.response = LA  
    session.userData.location = results.response  
    // session.userData.location = LA  
    builder.Prompts.text(session, 'How many days?')  
  },  
  function (session, results) {  
    // results.response = 5  
    session.userData.forecast = results.response  
    // session.userData.forecast = 5  
  
    // Find weather with info gathered  
    // Display results  
    session.endDialog()  
  }  
])  
)
```

Greeting Dialog



```
graph TD; A[Greeting Dialog] --> B[What is your name?]; B --> C[Hello Tierney]; C --> D[Weather Dialog]; E[ ] --> B; style E fill:none,stroke:none
```

A flowchart illustrating a sequence of four dialog steps. The steps are represented by rounded rectangular boxes arranged vertically. The first box is brown and contains the text 'Greeting Dialog'. A grey arrow points down from it to the second box, which is green and contains 'What is your name?'. A blue arrow points from the right into the second box. A grey arrow points down from the second box to the third box, which is green and contains 'Hello Tierney'. A grey arrow points down from the third box to the fourth box, which is brown and contains 'Weather Dialog'.

What is your  
name?

Hello Tierney

Weather Dialog

```
bot.dialog('/Greeting', [  
  function (session) {  
    builder.Prompts.text(session, 'What is your name?')  
  },  
  function (session, results) {  
    // results.response = Tierney  
    session.send('Hello', results.response)  
    // begin new dialog  
    session.beginDialog('/Weather')  
    // once Weather Dialog ends, return back to this line!  
    session.endDialog()  
  }  
])  
)
```

# Prompts & Inputs

Prompt Type	Description
<code>Prompts.choice</code>	Asks the user to choose from a list of choices.
<code>Prompts.digits</code>	Asks the user to enter a sequence of digits.
<code>Prompts.confirm</code>	Asks the user to confirm an action.
<code>Prompts.record</code>	Asks the record a message.
<code>Prompts.action</code>	Sends a raw <code>action</code> to the calling service and lets the bot manually process its outcome.



# CHATBOT BASICS

Builtin Prompts



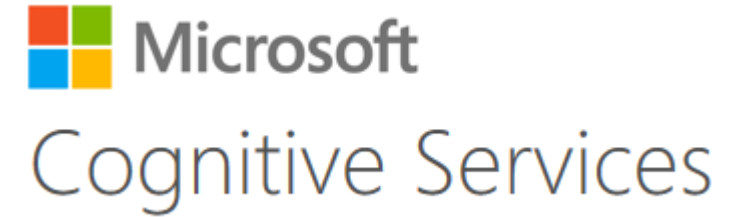


# CREATING INTELLIGENCE!

LUIS.ai

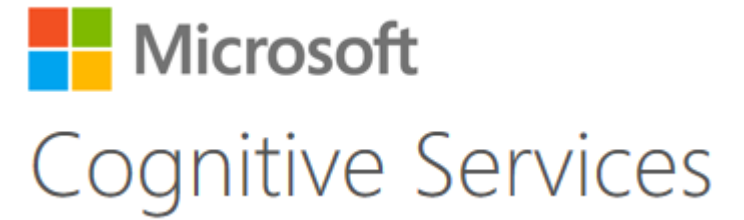


# LUIS.ai



- Language Understanding Intelligence Service (LUIS)
- Natural Language Processing (NLP)
  - *Using artificial intelligence to process natural language*
    - Extract Intents (meaning of the utterance)
    - Extract Entities (items in the utterance that are of value)
- This is a huge problem developers working with AI or human speech encounter
  - *Cognitive Services: LUIS abstracts this into a training model*

# LUIS.ai



Utterance:

‘I want to learn a lot and becoming inspired with  
**Breaking Into Bots** at **GOTO Conference!**’

Entities: **sessionTitle** | **conferenceTitle**

Intent: becomeInspired



# INCORPORATING INTELLIGENCE

LUIS.ai



# Thanks!

- [github.com/KSLHacks/BotScheduleDemodev.botframework.com](https://github.com/KSLHacks/BotScheduleDemodev.botframework.com)
- [luis.ai](https://luis.ai)
- [azure.microsoft.com/free](https://azure.microsoft.com/free) (\$200 credits)

// Kevin Leung

// Microsoft Technical Evangelist

// Twitter: **@KSLHacks**

// Github: **KSLHacks**

// Gabrielle Crevecoeur

// Microsoft Technical Evangelist

// Twitter: **@NoWaySheCodes**

// Github: **gcrev93**