PIC 16, Winter 2018

Lecture 6Wa: Intro to Amazon Alexa

Wednesday, February 14, 2018

Matt Haberland

Amazon Alexa

- Alexa is a *conversational* or *voice* (that is voice-based, rather than text-based or graphical) user interface (VUI)
- There are many Alexa-enabled devices available, including:
  - Speakers
  - Wristwatches
  - Phones
  - Thermostats

Echo Dot - ~$50

Echo - ~$180

- We're going to learn to write simple programs ("skills") for them using Python

What Can Alexa Do?

- Alexa has many built-in behaviors, like:
  - Setting a timer or alarm
  - Playing music/podcasts
  - Reading the news
  - Getting the weather
- Alexa also supports custom skills (think “apps”) created by developers.
- Some example custom skills:
  - Hello World – a very simple skill for demonstration purposes
  - Sergeant Tabata – times a “tabata” style workout

Why Amazon Alexa?

- Alexa is one of the first, and currently the most popular, voice-based interfaces
  - Getting a computer to “understand” and produce human speech are very difficult problems
  - We have to use an API (application programming interface) like Alexa’s to create VUI
    - While many programming languages support *GUI*s, no programming languages have built-in or standard VUI functionality (yet)
    - Writing our own system from scratch would be prohibitively difficult
- The API that developers can use to write Alexa skills is relatively advanced
  - Its closest competitor, Google Assistant Actions API, just became public at the end of last year
- This could be the next major disruption in computing

Get on board now!

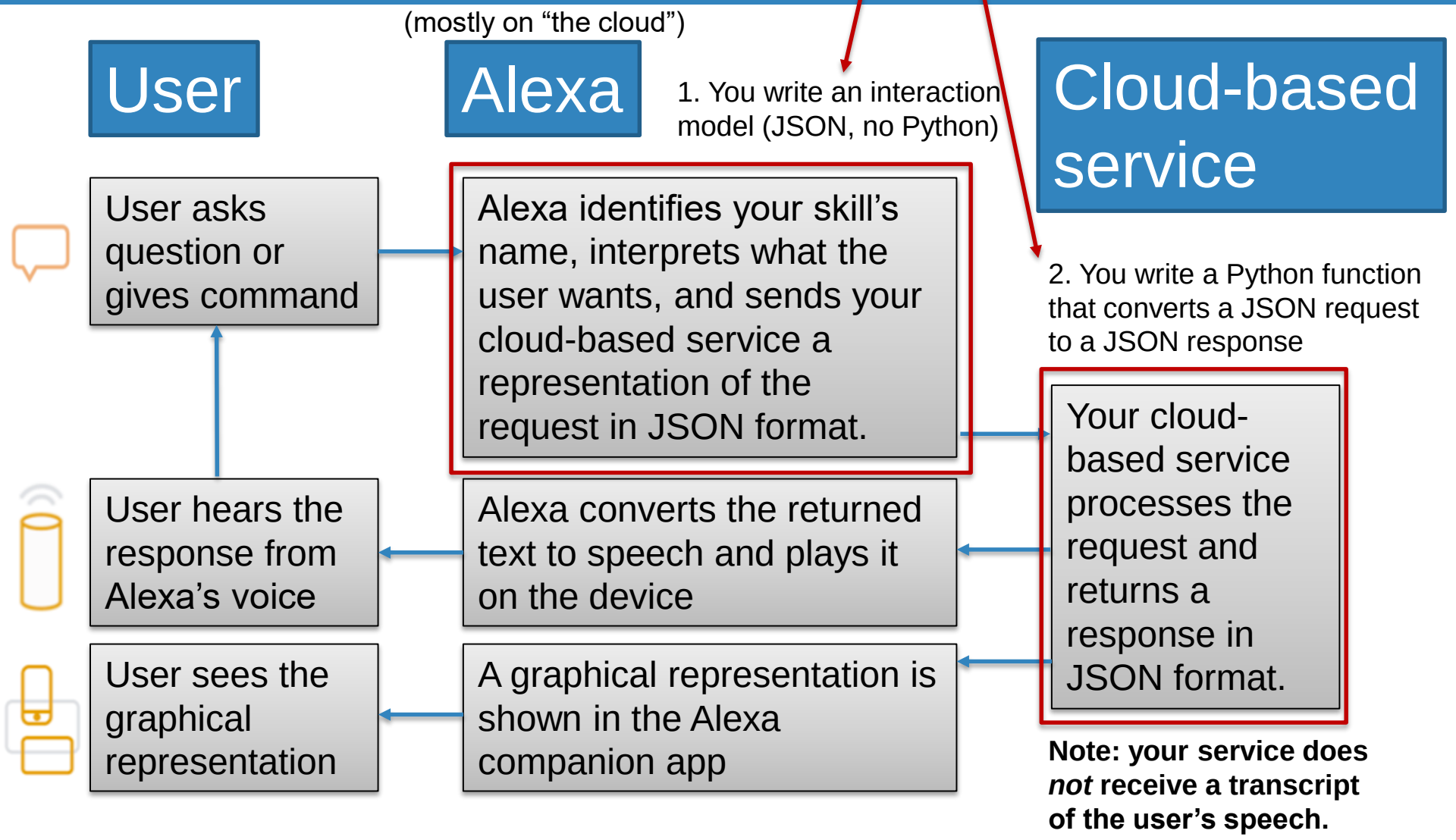
PIC 16, Winter 2018

Lecture 6Wb: Alexa Concepts and Terminology

Wednesday, February 14, 2018

Matt Haberland

Custom Skills



Adapted from: https://developer.amazon.com/public/solutions/alexa/alexa-skills-kit/overviews/understanding-custom-skills

Quiz

- True or false:
  - Your skill's code runs on the Alexa-enabled device
    - False. The device only recognizes the word "Alexa". All the rest of the audio data is processed on the cloud.
  - The only important part of an Alexa skill is the Python code
    - False. The code is half of the skill. You also need to write an interaction model.
  - Your Python code receives a transcript of the user's spoken words
    - False. It receives a request. Alexa can learn converts the user's spoken words to a request for your code based on the interaction model.
  - The two parts of your skill (interaction model and Python code) are both set up in the Alexa skill developer portal
    - False. Your code can be run on any cloud-based service. We'll use AWS Lambda (an Amazon service), but that is totally separate from the skill developer portal.

Invoking a Skill

"Alexa, start Sergeant Tabata"

Wake Word Supported word/phrase Skill Invocation Name

- Wake Word
  - Alexa is always listening, but only for her name
  - When she hears her name, she will attempt to understand words that follow
- Skill Invocation Name
  - The name of a skill that is spoken to start it or access its abilities
- Supported word/phrase
  - Starting words, like "ask", "launch", and "talk to", and connecting words, like "from", and "about", are supported to make starting a skill more natural

Intent vs No Intent

- There are two conceptually distinct ways in which the user can invoke a skill
- No Intent / `LaunchRequest` (just start a skill)

  "Alexa, start Sergeant Tabata"

  - The user simply wants to start the skill; they do not specify what they want the skill to *do*.
  - Typically, the skill will respond with a welcome message and potentially prompt the user for a more detailed request
- `IntentRequest` (specify what you want the skill to do)

  "Alexa, ask Sergeant Tabata to start a standard tabata"

  - The user starts a skill and tells it what they want it to do
  - As the designer of the skill, you will define different intents for which your skill will produce an appropriate response

Intents

- An intent is like a function in programming. It's something a user can ask your skill to do.
- Like functions, intents have names chosen by the programmer.

"Alexa, ask Sergeant Tabata to start a standard tabata"

This produces an `IntentRequest` with an intent called `StandardTabataIntent`

Utterances

- An *utterance* is the technical term for the user's vocal command/question/answer to Alexa
- Alexa (Amazon's servers) does the (difficult) job of mapping utterances to intents for a particular skill (and sending the skill the corresponding request)
- This is not trivial because the user can specify his/her intent using a variety of words/phrases (natural language), *not all of which need to be specified by skill's author*
- Utterances that map to the same intent:

Skill author specified that these should map to `HelloWorldIntent`

"Alexa, ask Hello World to say hello"

"Alexa, tell Hello World to say hi"

Alexa figured out that this should map to `HelloWorldIntent`

"Alexa, ask Hello World to greet me"

Interaction Model

- The interaction model is how you teach Alexa to map utterances to intents supported by your skill
- We'll see how this works (with examples) in much more detail later, but one important part is "sample utterances". For the Hello World skill:

```
HelloWorldIntent say hello
HelloWorldIntent say hello world
HelloWorldIntent hello
HelloWorldIntent say hi
HelloWorldIntent say hi world
HelloWorldIntent hi
HelloWorldIntent how are you
```

Session

- When you invoke an Alexa skill, you begin a *session*, which is a distinct conversation between you an Alexa
- During a session, you don't need to say the wake word to respond to Alexa's prompts
- The session ends when the skill indicates that the conversation is over, or when the user doesn't respond to a prompt from the skill
- Alexa can remember things within a session using *session attributes*
- Alexa *can't* remember things between sessions without the help of another service, like a database

We won't have time to cover this at all.
For my skills, I've used AWS DynamoDB

Session

User: *Alexa, start Sergeant Tabata*

Alexa: *Which would you like: a standard tabata, a custom tabata, or help?*

User: *Standard tabata*

Alexa: <leads tabata>

<session ends>

Session State Can't Affect Intent

- The *exact same* intent request will be generated whether the user:
  - First invokes the skill with no intent, then specifies an intent while the session is still open

    User: Alexa, start Hello World

    This produces a `LaunchRequest` (no intent)

    Alexa: Welcome to the Alexa Skills Kit, you can say hello

    User: Say hello

    Alexa: Hello world

  or

  - Invokes the skill and simultaneously specifies an intent

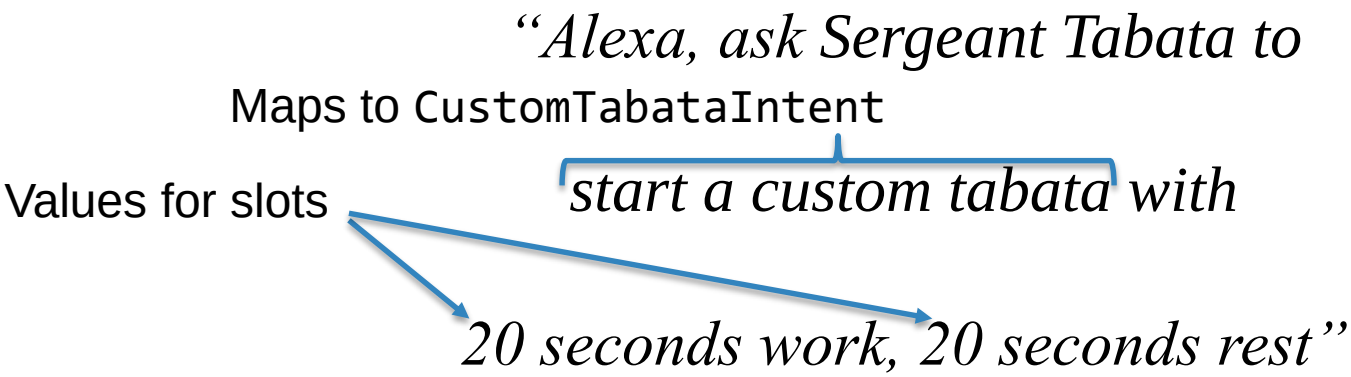
    User: Alexa, ask Hello World to say hello

    Alexa: Hello world

These produce *identical* `IntentRequests` with `HelloWorldIntent`

Slots

- As part of an utterance, the user can also pass values into intent parameters, which are called *slots*
- An intent can have multiple slots, but the user does not necessarily need to specify the values in order; Alexa determines which values go in which slots based on context



- Intents, like variables in C++, are typed
  - Amazon provides some types like AMAZON.NUMBER, AMAZON.DATE
  - You can also define custom types (lists of possible spoken words)

Request / Response

- Sessions consists of a series of *requests* and *responses*
  - When you are starting or are within a session, Alexa maps the utterance to an intent (based on your interaction model), and sends a JSON representation to your cloud-based service called a *request*.
  - Your cloud-based service (a Python function) processes the request and produces a JSON *response*, which Alexa (in conjunction with the Alexa-enabled device) converts into audible speech.

Request

```
{
  "session": {
    "sessionId": "SessionId.748bbf41-aea3-495d-82f4-e53052
    "application": {
      "applicationId": "amzn1.ask.skill.ca7cc50a-1303-4321
    },
    "attributes": {},
    "user": {
      "userId": "amzn1.ask.account.AEFDSXHN3U3PSYQVOGENS7C
    },
    "new": true
  },
  "request": {
    "type": "LaunchRequest",
    "requestId": "EdwRequestId.7863f709-ac84-434f-8d40-a39
    "locale": "en-US",
    "timestamp": "2017-06-06T20:41:24Z"
  },
  "version": "1.0"
}
```

Response

```
{
  "version": "1.0",
  "response": {
    "outputSpeech": {
      "type": "PlainText",
      "text": "Welcome to the Alexa Skills Kit, you can say hello"
    },
    "card": {
      "content": "Welcome to the Alexa Skills Kit, you can say hello",
      "title": "HelloWorld",
      "type": "Simple"
    },
    "reprompt": {
      "outputSpeech": {
        "type": "PlainText",
        "text": "Welcome to the Alexa Skills Kit, you can say hello"
      }
    },
    "shouldEndSession": false
  },
  "sessionAttributes": {}
}
```

Your cloud-based service does exactly one thing: read a request and write an appropriate response