PIC 16, Winter 2018 – Preparation 6W

Assigned 2/9/2018. To be completed by class 2/14/2018.

Intended Learning Outcomes

By the end of this preparatory assignment, students should be able to:

- explain the difference between a physical Alexa-enabled device (like the Echo) and Amazon's Alexa service,
- follow explanations that include new terms like voice user interface, skill, invocation name, intent, utterance, interaction model, cloud based service, wake word, request, and response, and
- perform simple verbal interactions with an Alexa-enabled device.

Tasks

- ☐ Watch this if you don't know what an Amazon Echo is.
- ☐ Watch this for more information about Alexa.
- ☐ Optionally, skim this for more reason to learn to program Alexa skills.
- ☐ Get some experience interacting with Alexa. Visit https://echosim.io/, log in with an Amazon account, and try out some of these suggestions.
- ☐ Designing a good *voice user interface* (VUI) is a bit less intuitive than designing a passable GUI. While the design process and considerations are very important, it's outside the scope of the class. In this class, I'll prescribe the interface, your job will be to code it. If you decide to pursue VUI programming beyond this class, Amazon has some great resources starting here.
- ☐ Read the top paragraph of Understanding Custom Skills.
- ☐ Read "Components of a Custom Skill". There's a fair amount of new terminology here. It's essential to understand these terms in order to continue, so I'll give my own definitions and examples to reinforce the ones in the reading:
    - A *skill* is like a program or app. However, whereas an app on your cell phone or computer usually has a GUI, a skill has a VUI.
    - The *invocation name* is what the user will say to tell Alexa that he/she wants to use your skill. Saying the invocation name is like clicking an icon to start an app.
    - A single skill may support many *intents*, or functions a user will want to accomplish with your skill. For instance, a skill for playing a podcast might have intents for playing an episode, pausing, resuming from pause, skipping to the next episode, and getting help for using the skill. These intents correspond quite neatly with the buttons and menu items you might find in a podcast player app. In some ways, an intent is similar to a function (in the programming sense), but this does not necessarily mean that you will write a different function for every intent.
    - An *utterance* is what the user will say to specify an intent. To allow for a more natural user experience, many utterances can map to (connect to / be interpreted to refer to) the same intent. For instance, "start episode 9", "play show number 9", and "I want to listen to 9" are all utterances that might map to an intent that plays a particular episode of a podcast.
    - The first part of creating a skill is to configure in Amazon's "developer portal" an *interaction model* for your skill. This includes specifying a few *sample utterances* to help Alexa learn to map a user's utterance to an intent supported by your skill. Alexa will try to generalize from your sample utterances; even if the user says something a little different from *all* your examples, the utterance may still achieve the desired intent. Note: configuring the skill does not require any Python programming.

    - The second part of creating a skill is to configure a *cloud-based service* (program running on a computer connected to the internet). This is where Python comes in: your cloud-based service will be a Python program running on an Amazon server. Based on your interaction model, Alexa will map the user's utterance to an intent and communicate this to your cloud based service. Your Python program will perform whatever actions the intent requires, which likely includes providing information to be communicated back to the user.
- Here is my own summary of what happens when a user interacts with an Alexa-enabled device like the Echo. I'll use the terminology from above and introduce some new terms (italicized). Let's say you've created a skill called "Python Podcast".
    - The user speaks a phrase like "Alexa, ask Python Podcast to play episode one."
    - The Alexa-enabled device does little more than recognize the *wake word*, "Alexa". If the wake word is spoken, the rest of the sentence audio data is sent over the internet to *Alexa*, the cloud service created by Amazon that recognizes the spoken words.
    - Alexa (Amazon's cloud service) determines that the user is attempting to use the "Python Podcast" skill based on the invocation name "Python Podcast".
    - Based on the interaction model that has been configured in the developer portal for "Python Podcast", Alexa maps the user's utterance to an intent for playing a podcast episode.
    - Alexa sends a *request* specifying the intent to the cloud-based service configured for Python Podcast. A request is simply plain text in JSON format that your Python program can interpret as a Python dictionary.
    - Your Python program analyzes the contents of the request dictionary. Based on the intent, your program prepares another Python dictionary. This is converted to a plain-text, JSON-formatted *response* to Alexa with instructions for what audio should be played to the user (and what, if any, graphical representation of the response should be presented to the user in the Alexa mobile app).
    - Alexa sends audio data to the user's Alexa-enabled device (and sends graphics and text to the Alexa mobile app)
    - The Alexa-enabled device plays the audio to the user.
- Skim Conducting a Conversation with the User. This introduces the idea that Alexa skills can carry on a conversation with the user. However, in this class, we will only specifically discuss *one-shot* skills in which the user speaks a phrase and the Alexa-enabled device speaks a single response. You will be prepared to learn on your own about other possibilities when we're done.
- Skim Providing a Visual Component for Your Skill. We won't cover how to do this specifically, but it's not difficult once you understand how to provide audio responses.
- Hosting the Cloud-Based Service for Your Skill introduces Amazon Web Services (AWS) Lambda. AWS Lambda is the cloud-based service that will run the Python program at the heart of your skill. AWS Lambda is not just for Alexa skills. It's a totally separate service from Amazon that is useful for:
    - executing "on the cloud"
    - a function written in C##, Java, JavaScript, or Python, that must
    - accept a plain-text, JSON-formatted request, and
    - return a plain-text, JSON-formatted response.

  Since it's also offered by Amazon, it's a very convenient choice for our cloud-based service. We will configure our Alexa skill in the developer portal to invoke our Python code on AWS Lambda.

  Believe it or not, there is more terminology to come. But to be honest, once you understand 1) the terminology, 2) the JSON format for requests, and 3) the JSON format for the response your Python program must return, writing a skill in Python is very easy. We'll learn more in class!