MAKING YOUR OWN ALEXA SKILL

SHRIMAI PRABHUMOYE, ALAN W BLACK

WHAT IS ALEXA?

- **Alexa** is an intelligent personal assistant developed by Amazon.
- It is capable of voice interaction, music playback, making to-do lists, setting alarms, streaming podcasts, playing audiobooks, and providing weather, traffic, and other real time information, such as news.
- Alexa can also control several smart devices using itself as a home automation system.

INSIDE ALEXA

amazon echo

Volume ring

Reflex port
Enhances the woofer's output for deeper sounds without distortion

2.5 inch woofer
Delivers deep bass response

2.0 inch tweeter
Crisply hits the high notes

9.25 in

3.27 in

Light ring

Action button

7-microphone array

Microphone off button

Power adapter

Echo also includes a remote with a built-in microphone and music playback and volume controls.

ALEXA SKILL

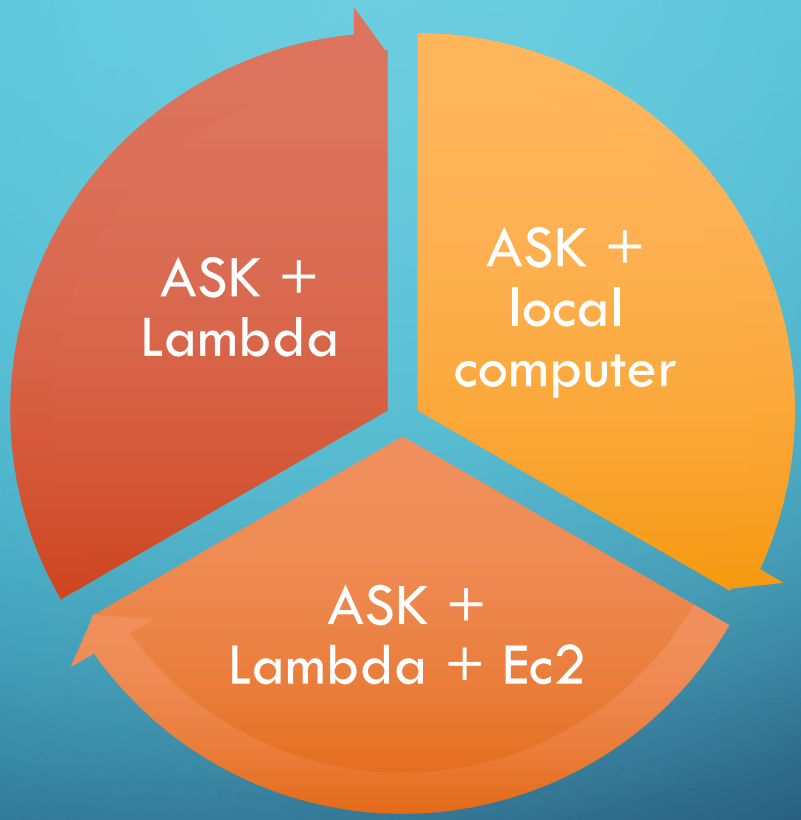
- Alexa skills are like apps.
- You can enable and disable skills, using the Alexa app.
- Skills are voice-driven Alexa capabilities
- Alexa Skills Kit (ASK) is a collection of self-service APIs, tools, documentation, and code samples.
- We will use ASK to create a skill, define intents, define slots and connect to our python program.

INTENTS AND SLOTS

- Intents: actions that the user wants the system to perform.
- Slots: possible types of actions
- Sample Utterrances: A set of likely spoken phrases mapped to the intents.

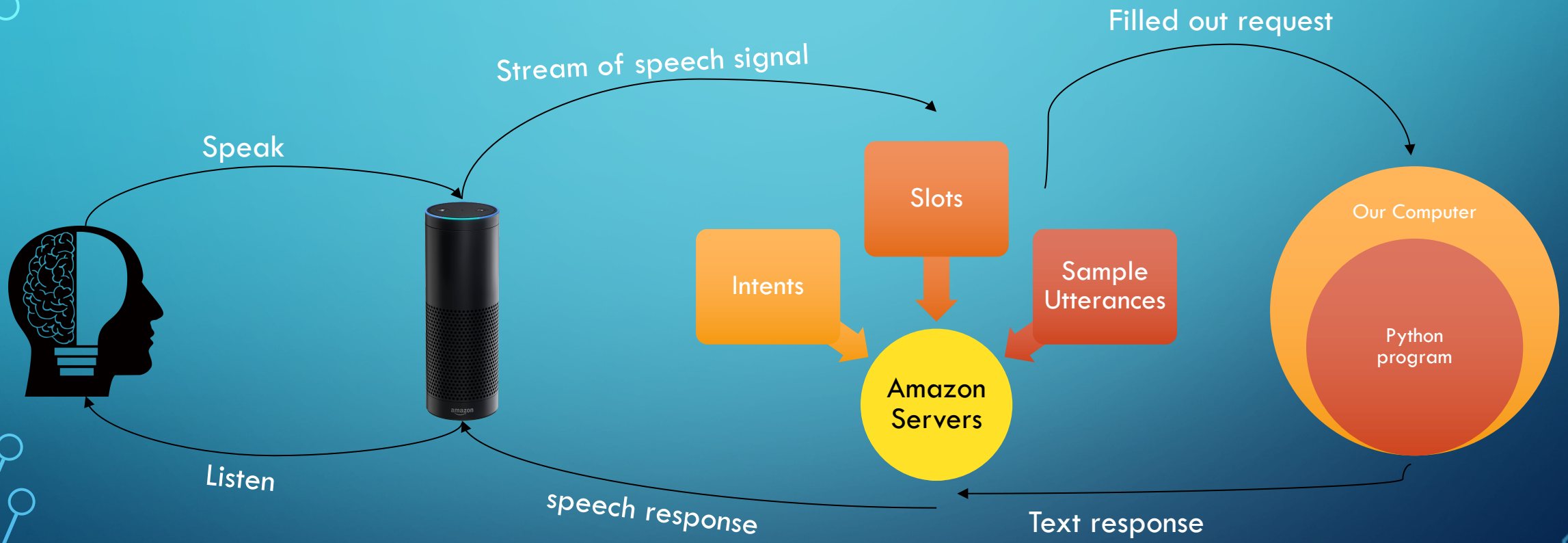
Intent	Slots	Sample Utterances
Open	Door, jar	Open door Open jar Open the door Open the jar
Yes	-	Yes Yeah Sure Agreed

HOW TO BUILD A SKILL



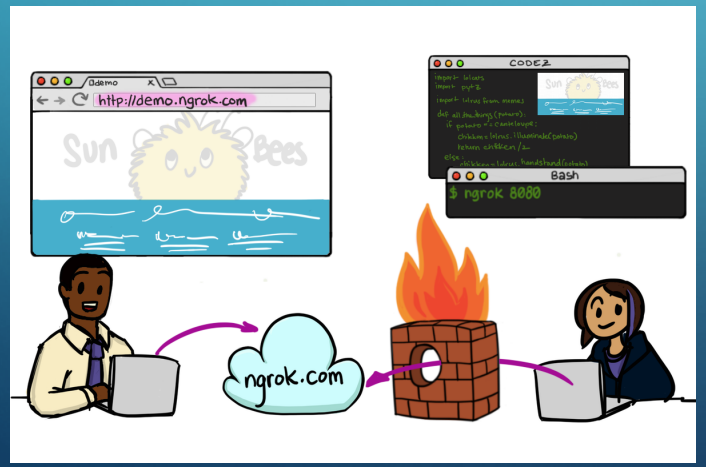
- Note: Lambda is an event-driven, serverless computing platform provided by Amazon as a part of the Amazon Web Services.

HOW WE WILL BUILD A SKILL



FLASK-ASK AND NGROK

- We will create a Alexa skill using Flask-ask and ngrok.
- Flask-Ask is a Flask Extension.
  - Helps construct ask responses.
  - Has decorators to map Alexa requests and intent slots to view functions.
  - Makes session management easy.
- Ngrok lets you expose a local server behind a NAT or firewall to the internet.



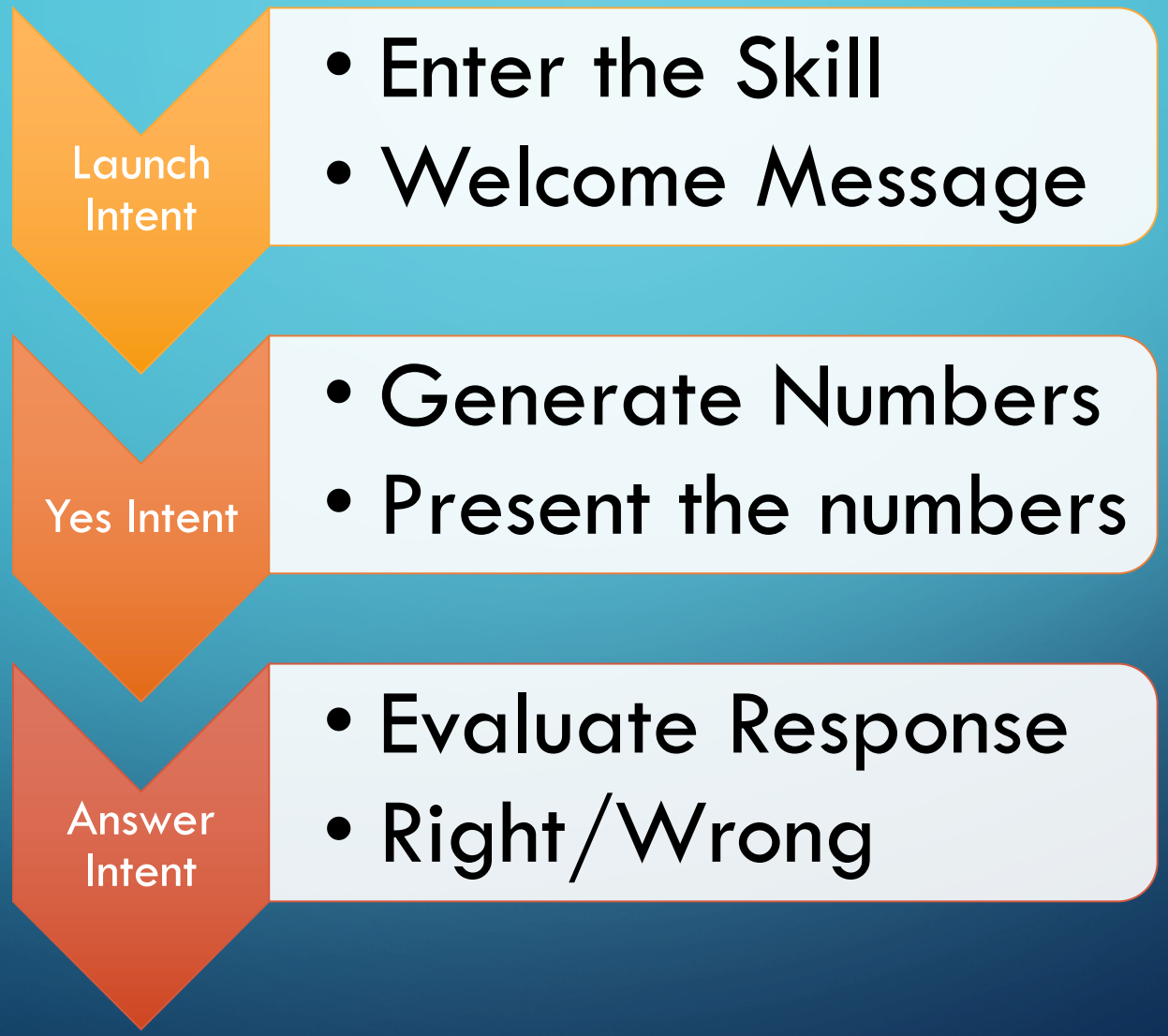
SIMPLE EXAMPLE

```
from flask import Flask
from flask_ask import Ask, statement

app = Flask(__name__)
ask = Ask(app, '/')

@ask.intent('HelloIntent')
def hello(firstname):
    speech_text = "Hello %s" % firstname
    return statement(speech_text).simple_card('Hello',
speech_text)

if __name__ == '__main__':
    app.run()
```

MEMORY GAME

- The computer agent will generate 3 random integers and will ask you to repeat the three numbers in the reverse order.
- If you memorize the numbers correctly and repeat them in the reverse order, you are right otherwise you are wrong.

MEMORY GAME


Launch Intent
• Enter the Skill
• Welcome Message
Yes Intent
• Generate Numbers
• Present the numbers
Answer Intent
• Evaluate Response
• Right/Wrong

MEMORY GAME

- Flask-Ask lets you separate code and speech with templates.
- Create a memory_game.py
- Create a file named *templates.yaml* in the same location as *memory_game.py*
- *On our computer, we run memory_game.py on one terminal and*

./ngrok http 5000

on another terminal and keep the connection open.

MEMORY GAME

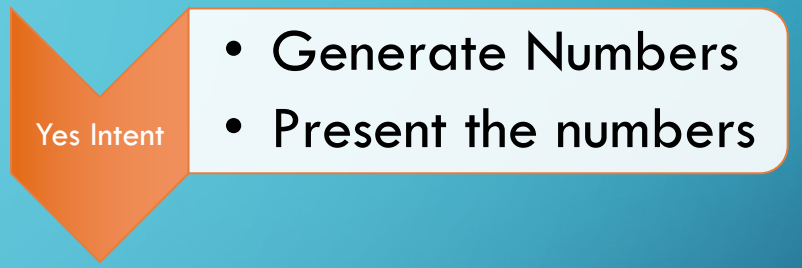
Launch Intent

- Enter the Skill
- Welcome Message

```
@ask.launch
def new_game():
    welcome_msg = render_template('welcome')
    return question(welcome_msg)
```

```
{welcome: "Welcome to memory game. I'm going
to say three numbers for you to repeat
backwards. Ready?”}
```

MEMORY GAME




```
@ask.intent("YesIntent")
def next_round():
    numbers = [randint(0, 9) for _ in range(3)]
    round_msg = render_template('round', numbers=numbers)
    session.attributes['numbers'] = numbers[::-1]
    return question(round_msg)
```

```
{round: "Can you repeat the numbers {{ numbers |
join(", ") }} backwards?”}
```

MEMORY GAME

```
@ask.intent("AnswerIntent",
convert={'first': int, 'second':
 int, 'third': int})
def answer(first, second, third):
    winning_numbers = session.attributes['numbers']
    if [first, second, third] == winning_numbers:
        msg = render_template('win')
    else:
        msg = render_template('lose')
    return statement(msg)
```

Answer Intent

- Evaluate Response
- Right/Wrong

```
{ win: "Good job!",
lose: "Sorry, that's the wrong answer."}
```

MEMORY GAME - INTENTS

```
{
    "intents": [{
        "intent": "YesIntent"
          },{
        "intent": "AnswerIntent",
        "slots": [{
              "name": "first",
              "type": "AMAZON.NUMBER"
         }, {
              "name": "second",
              "type": "AMAZON.NUMBER"
         }, {
              "name": "third",
              "type": "AMAZON.NUMBER"
         }]
    }]
}
```

SLOT TYPES

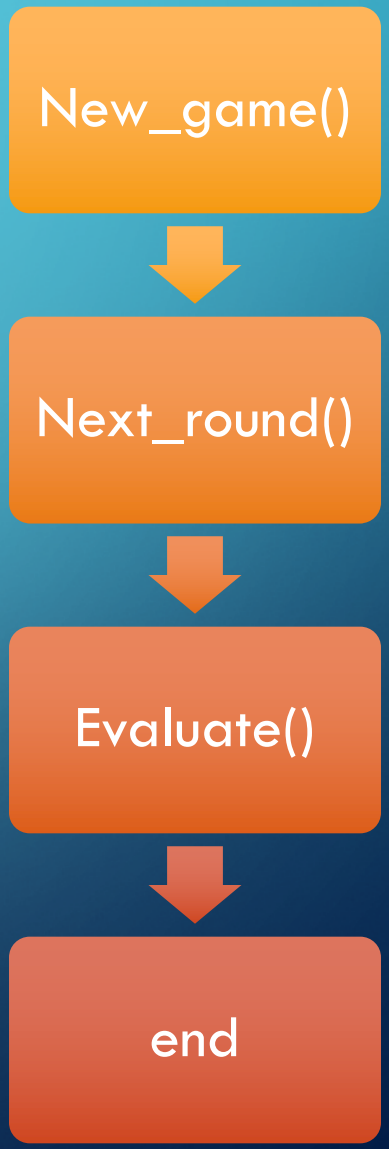
- Amazon has some built-in slot types:
  - AMAZON.DATE
  - AMAZON.DURATION
  - AMAZON.NUMBER
  - AMAZON.TIME
- For identifying strings, we define a custom slot type, say PizzaToppings.
- We then give all possible values we want to be identified as PizzaToppings
- You can add multiple custom slot types.

MEMORY GAME - LANGUAGE

Yes Intent	Answer Intent
• Yes • Sure	• {first}{second}{third} • {first}{second} and {third}

SKILL AS A GRAPH

- A skill can be represented as graph
- A session consists of the "conversation" moving from state to state
- The transitions between states are taken on the basis of what the human speaks
- Spoken input is an "intent" and is typically prompted by a question spoken by the system
- Together this describes the conversation that a skill can handle



STEPS TO BUILD A SKILL

- Define what the skill should do, its purpose/goal.
- What are some interactions/scenarios you'd handle?
- Mock up the system and try it! Does it work like you thought it would?
- Define the intents required by your skill. (Intents and Slots)
- Brain storm the language you expect people to use. (Sample Utterances)
- Figure out the language Alexa will use. (template.yaml)
- Construct the Action part of the skill

PRESENTATION

- Idea: need for your idea
- Scenarios you can handle
- Intents
- Slots
- Sample Utterances
- Failures of the skill
- Demo
- What you learnt from workshop.

USEFUL LINKS

- https://developer.amazon.com/blogs/post/Tx14R0IYYGH3SKT/flask-ask-a-new-python-framework-for-rapid-alexa-skills-kit-development
- https://github.com/johnwheeler/flask-ask