Lab 3: Building Space Geek

Welcome to Lab 3!

In this lab, we are going to build a simple Alexa Skill named Space Geek, which gets Alexa to tell us space facts. This Skill has no external dependencies or session management (we will discuss what this means later), and it shows the most basic example of how to create a Lambda function for handling Alexa Skill requests.

Overview

There are two parts to an Alexa skill. The first part is the Voice User Interface (VUI). This is where we define how we will handle a user's voice input, and which code should be executed when specific commands are uttered. The second part is the actual code logic for our skill, which contains the code to be executed when these commands are uttered.

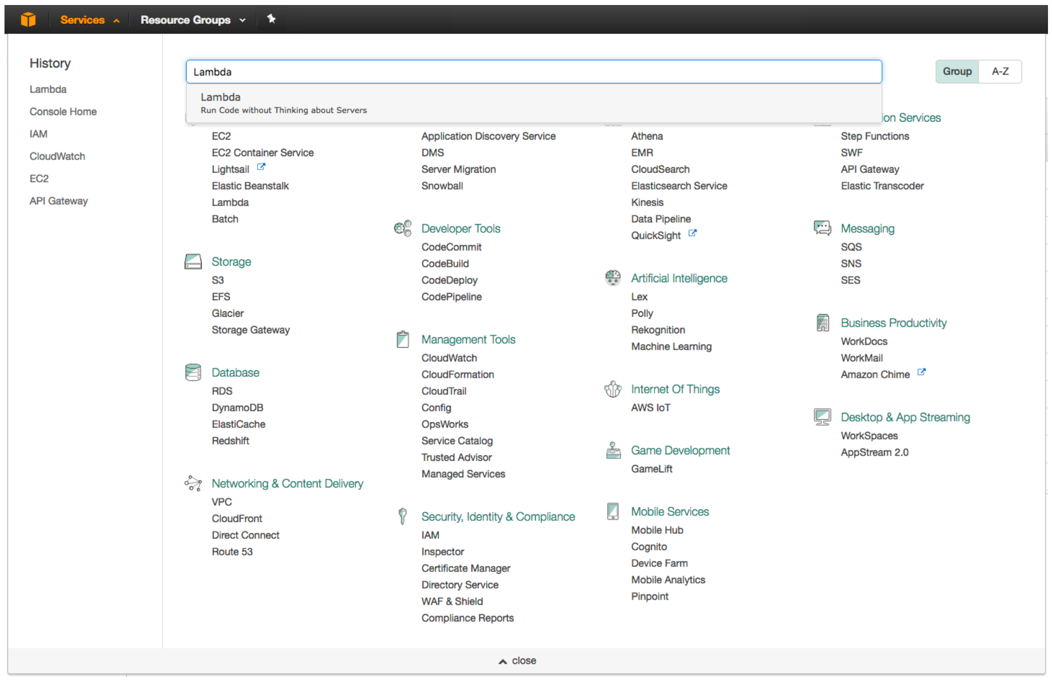
In this lab, we will start by configuring the code logic using the step-by-step guide below.

Part I:

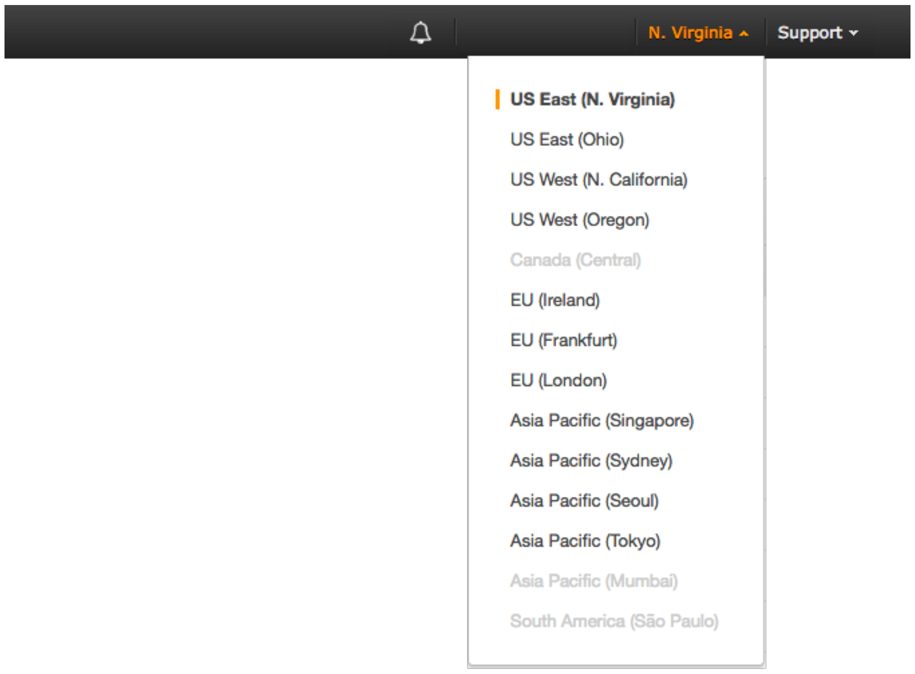
AWS Lambda Setup

1. **Go to http://aws.amazon.com and sign in to the Amazon Developer console.** If you don't already have an account, you will need to create one. Check out this quick walkthrough for setting up a new AWS account.

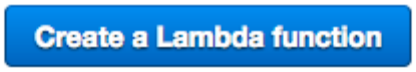
2. **Choose "Services" at the top of the screen, and type "Lambda" in the search box.** You can also find it in the list of services. It is in the "Compute" section.



3. **Check your AWS region.** Lambda only works with the Alexa Skills Kit in two regions: US East (N. Virginia) and EU (Ireland). We want to choose the region closest to our customers – in this case, US East (N. Virginia).

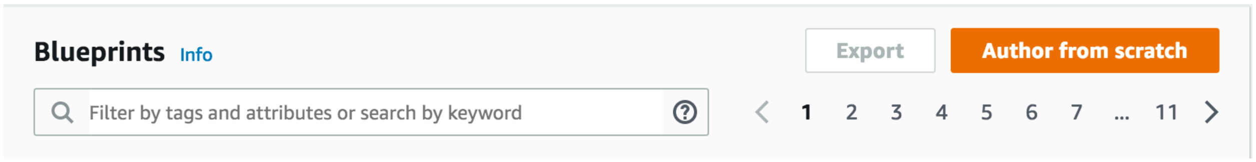


4. **Click the "Create a Lambda function" button.** It should be near the top of your screen.

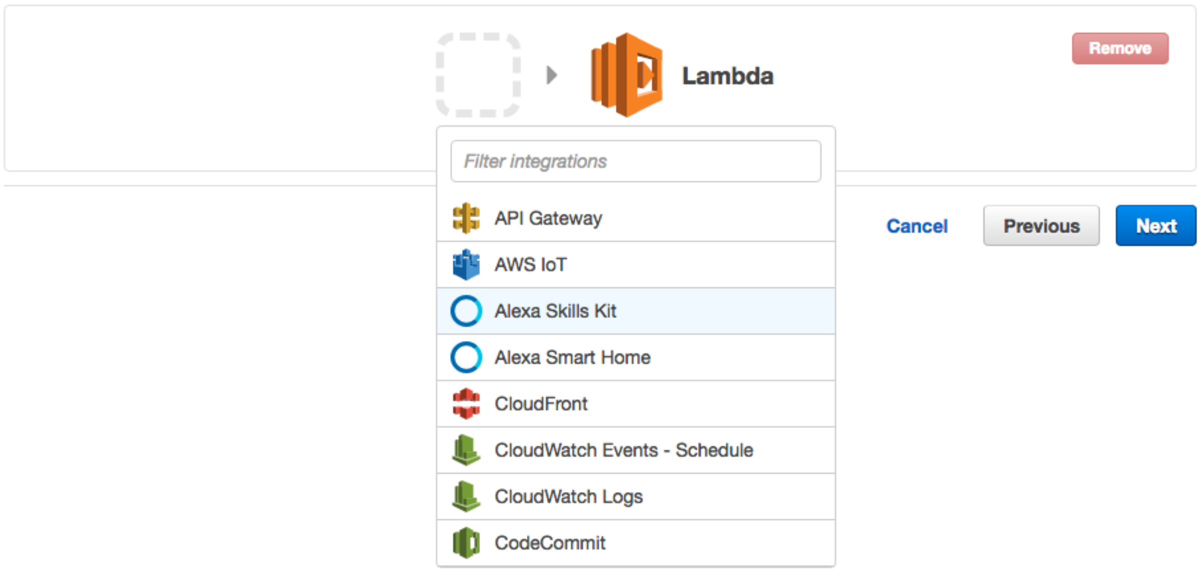


5. **Select "Author from scratch" for the blueprint.** Blueprints are shortcuts to getting everything set up for your skill. It adds the alexa-sdk to your Lambda function so that you don't have to upload it yourself. However, AWS currently only provides blueprints for skills being implemented in Python or Node.js.

   Because we are building a skill from scratch in Java, we don't want to use one of these preconfigured blueprints, and will thus choose to "Author from scratch."



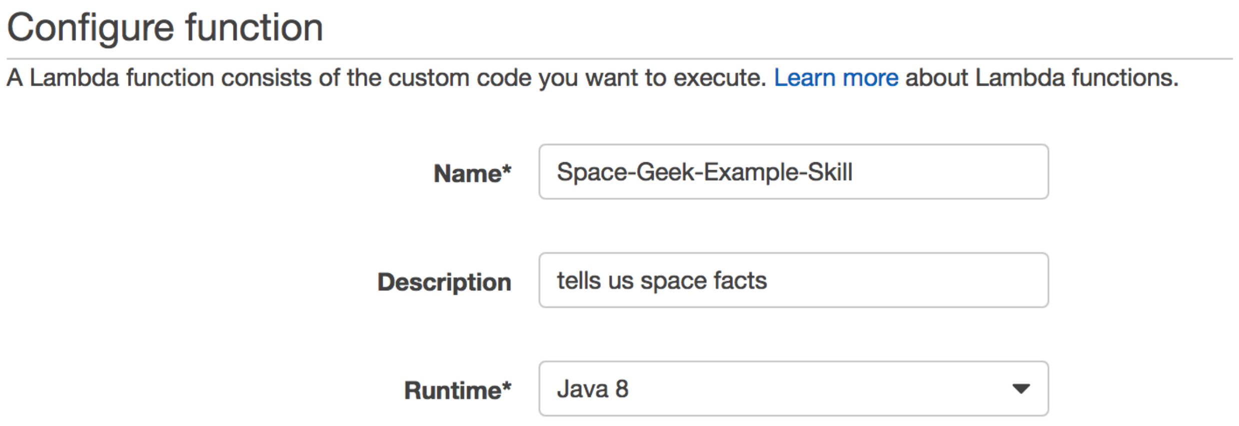
6. **Configure your trigger.** There are many different AWS services that can trigger a Lambda function, but for the purposes of this exercise, we need to select "Alexa Skills Kit." If you don't see Alexa Skills Kit in the list, check your AWS Region in the top right corner.



Once you have selected Alexa Skills Kit, click the **Next** button.

7. **Configure your function.** This screen is where we will enter the important parts of our Lambda function. These values will only ever be visible to you, but make sure that you name your function something meaningful. "Space-Geek-Example-Skill" is sufficient if you don't have another idea for a name. Go ahead and enter a description of your choice.

8. Select "**Java 8**" as the Runtime.



Configure function

A Lambda function consists of the custom code you want to execute. Learn more about Lambda functions.

Name* Space-Geek-Example-Skill

Description tells us space facts

Runtime* Java 8

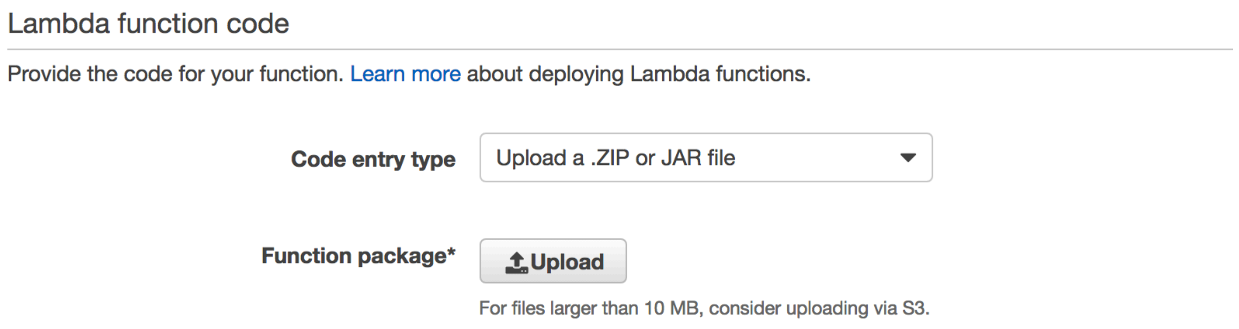
9. **Build the project.** Open up Terminal / Command Prompt, and navigate to the `alexa-skills-kit-java/samples` directory, and run the following command:

   ```
   mvn assembly:assembly -DdescriptorId=jar-with-dependencies package
   ```

   This will generate a zip file named "alexa-skills-kit-samples-1.0-jar-with-dependencies.jar" in the target directory.

   [If you're having trouble navigating to the directory, open up Github Desktop, right click the alexa-skills-kit-java repository, and click "Open in Terminal / Command Prompt."]

10. **Upload the JAR file.** In the AWS Console in your browser, select Code entry type as "Upload a .ZIP file."



Lambda function code

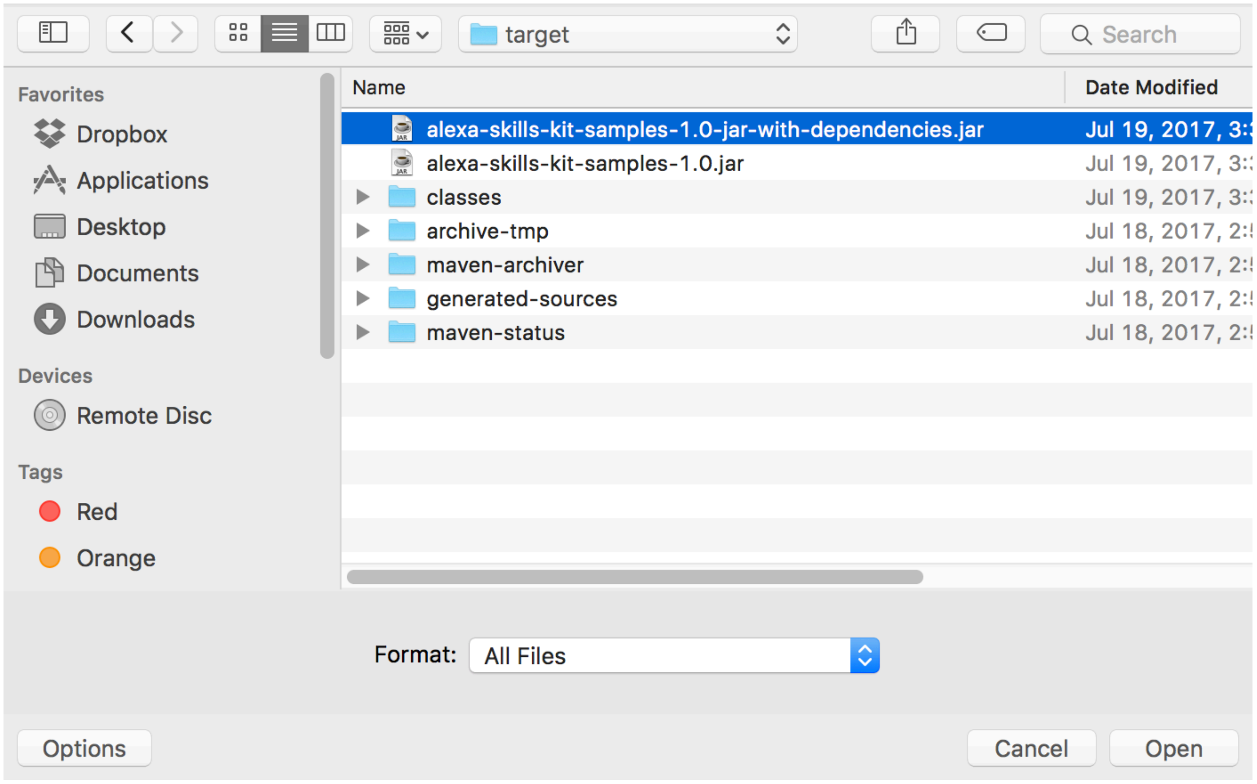
Provide the code for your function. Learn more about deploying Lambda functions.

Code entry type Upload a .ZIP or JAR file

Function package*

For files larger than 10 MB, consider uploading via S3.

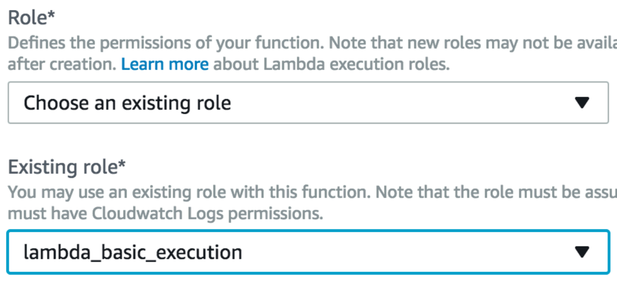
11. Then upload the "alexa-skills-kit-samples-1.0-jar-with-dependencies.jar" file from the build directory to Lambda.



Name	Date Modified
alexa-skills-kit-samples-1.0-jar-with-dependencies.jar	Jul 19, 2017, 3:
alexa-skills-kit-samples-1.0.jar	Jul 19, 2017, 3:
classes	Jul 19, 2017, 3:
archive-tmp	Jul 18, 2017, 2:
maven-archiver	Jul 18, 2017, 2:
generated-sources	Jul 18, 2017, 2:
maven-status	Jul 18, 2017, 2:

12. Set the Handler as spacegeek.SpaceGeekSpeechletRequestStreamHandler (this refers to the Lambda RequestStreamHandler file in the jar)

13. **Set up your Lambda function role.** If you haven't done this before, here is a detailed walkthrough for setting up your first role for Lambda. If you have done this before, you only need to set your **Existing role** value to "lambda_basic_execution."



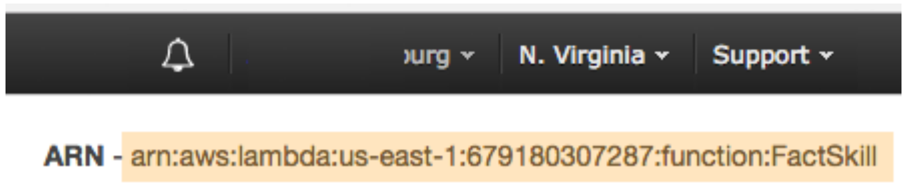
Role*
Defines the permissions of your function. Note that new roles may not be available for a few minutes after creation. Learn more about Lambda execution roles.
Choose an existing role

Existing role*
You may use an existing role with this function. Note that the role must be assumable by Lambda and must have Cloudwatch Logs permissions.
lambda_basic_execution

14. **You can skip all of the Tags and Advanced settings.** Simply click the **Next** button to move to the Review screen.

15. **The Review screen is just a summary of your choices. Click the Create Function button in the bottom right corner.**

16. **As a final step, copy the ARN value from the top right corner of the screen.** You will need this value in the next section of this guide.



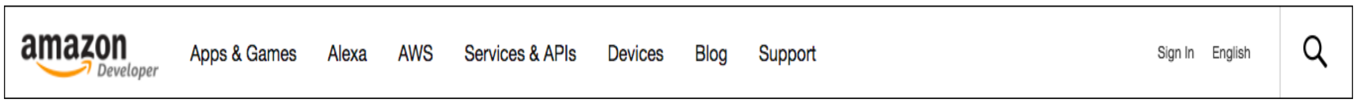
ARN - arn:aws:lambda:us-east-1:679180307287:function:FactSkill

Part II

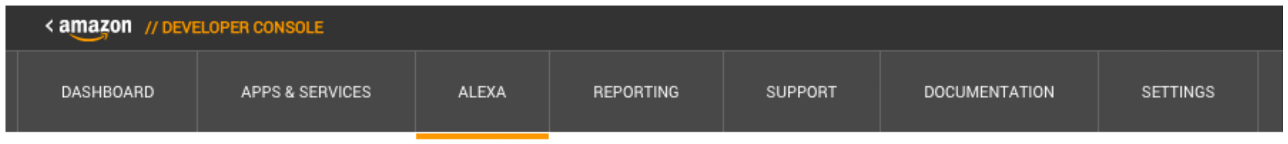
Alexa Skill Setup

Now let’s return to the front end of the Alexa skill – configuring the Interaction Model. Again, this is where we define how we will handle a user's voice input, and which code should be executed when specific commands are uttered.

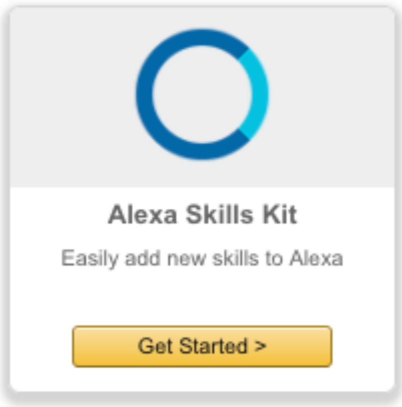
1. **Go to the [Amazon Developer Portal](). In the top-right corner of the screen, click the "Sign In" button.**
   (If you don't already have an account, you will be able to create a new one for free.)



2. **Once you have signed in, click the Alexa button at the top of the screen.**

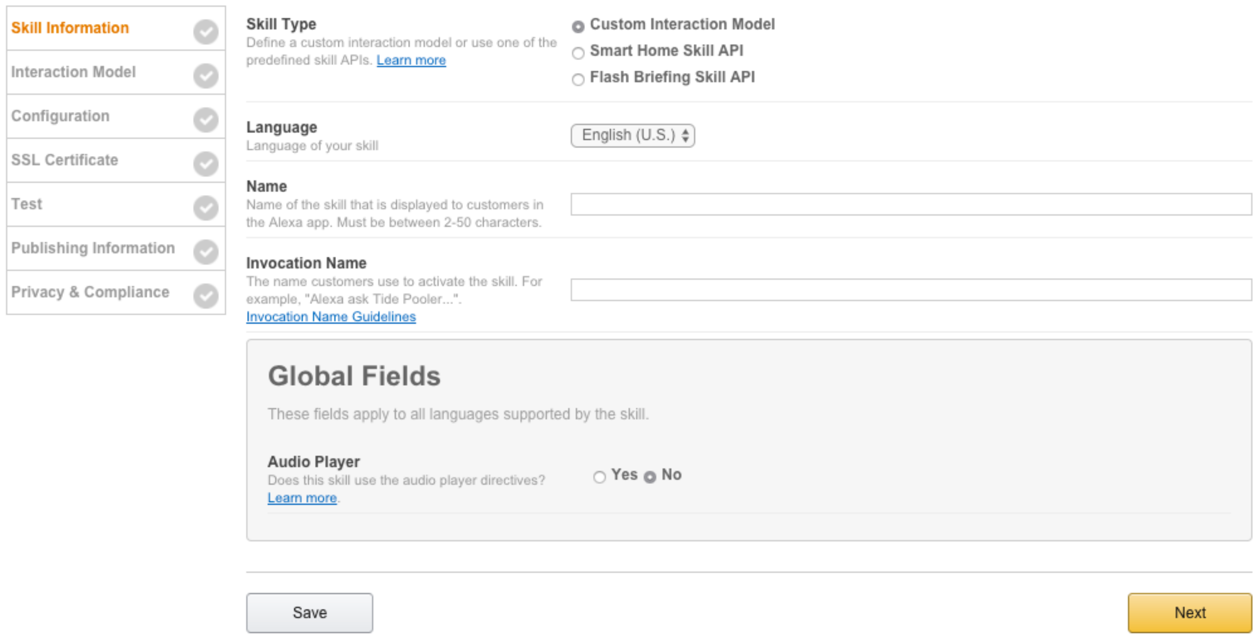


3. **On the Alexa page, choose the "Get Started" button for the Alexa Skills Kit.**



4. **Select "Add A New Skill."** This will get you to the first page of your new Alexa skill.

5. **Fill out the Skill Information screen.**
   Set "SpaceGeek" as the skill name and "space geek" as the invocation name. The invocation name is what is used to activate your skill. For example you would say: "Alexa, ask Space Geek for a space fact."
   Make sure to review the tips provided below the screenshot.



Skill Type
Define a custom interaction model or use one of the predefined skill APIs. Learn more
[✓ Custom Interaction Model]
Smart Home Skill API
Flash Briefing Skill API

Language
Language of your skill
English (U.S.)

Name
Name of the skill that is displayed to customers in the Alexa app. Must be between 2-50 characters.

Invocation Name
The name customers use to activate the skill. For example, "Alexa ask Tide Pooler...".
Invocation Name Guidelines

Global Fields
These fields apply to all languages supported by the skill.

Audio Player
Does this skill use the audio player directives?
Learn more.
Yes [✓ No]

Skill Information Tips

a. **Skill Type** For this exercise, we are creating a skill using the Custom Interaction Model. This is the default choice.

b. **Language** Choose the first language you want to support. You can add additional languages in the future, but we need to start with one. (This guide is using U.S. English to start.)

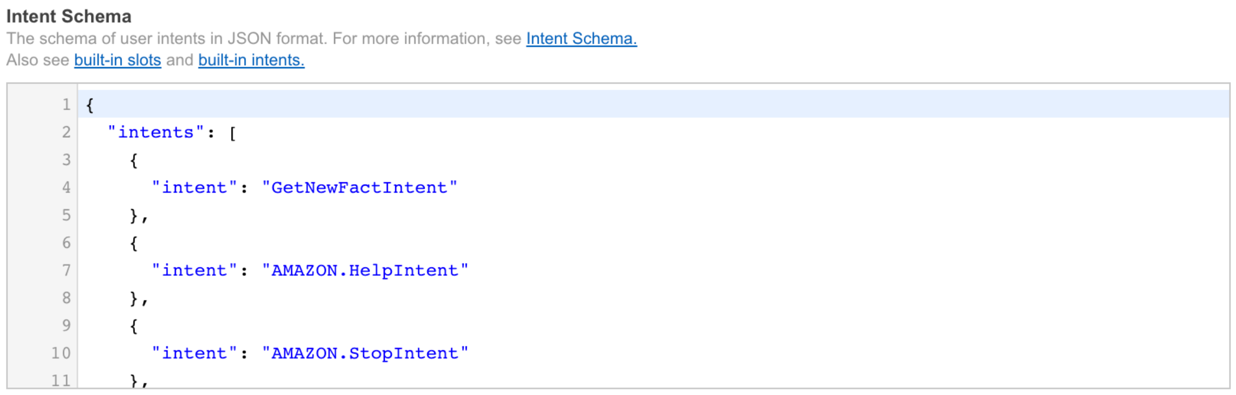
c. **Name** This is the name that will be shown in the Alexa Skills Store, and the name your users will refer to.

d. **Invocation Name** This is the name that your users will need to say to start your skill. You can review the entire set of [Invocation Name Requirements] here.

6. **Click the Next button to move to the Interaction Model.**

7. **Define the Intent Schema.** An intent represents an action that fulfills a user's spoken request.

   Copy and paste the contents of the speechAssets/IntentSchema.json file included. There should be four intents: (1) get a new fact, (2) help, (3) stop, and (4) cancel.

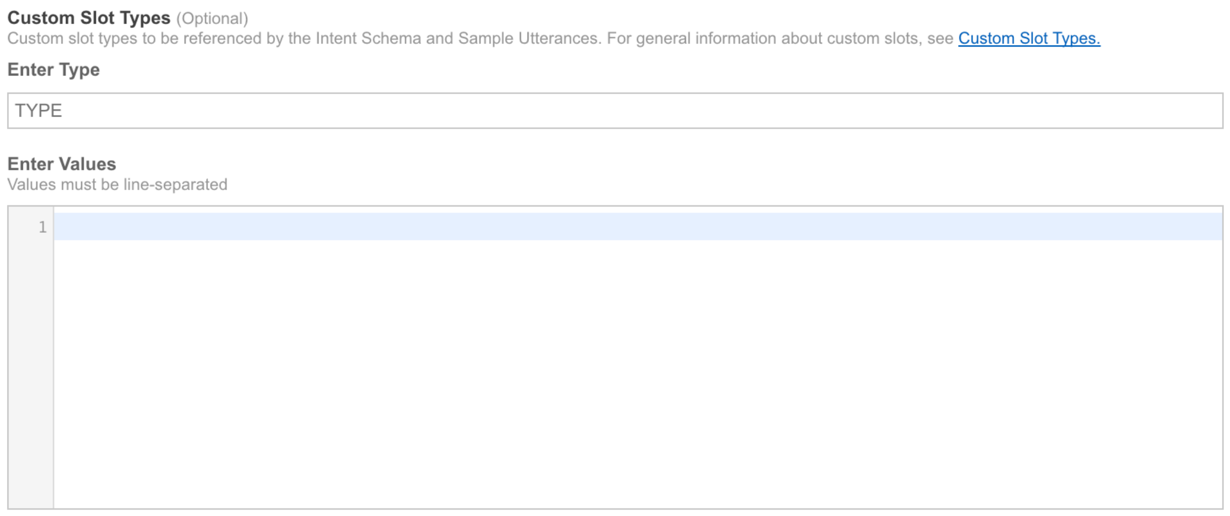


Intent Schema

The schema of user intents in JSON format. For more information, see Intent Schema.
Also see built-in slots and built-in intents.

```
{
  "intents": [
    {
      "intent": "GetNewFactIntent"
    },
    {
      "intent": "AMAZON.HelpIntent"
    },
    {
      "intent": "AMAZON.StopIntent"
    },
```

8. **Leave Custom Slot Types blank.** We will learn what custom slots are later.


Custom Slot Types (Optional)
Custom slot types to be referenced by the Intent Schema and Sample Utterances. For general information about custom slots, see Custom Slot Types.

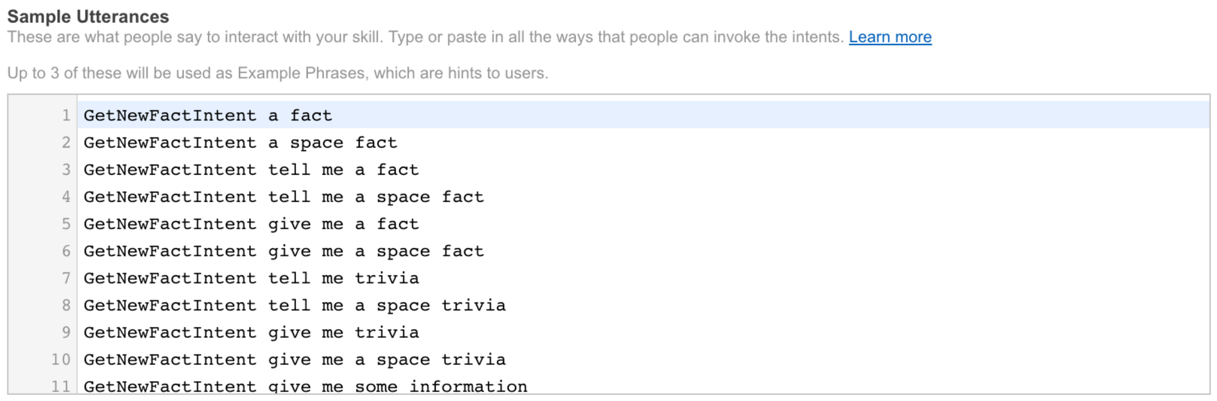
Enter Type

TYPE

Enter Values
Values must be line-separated

9. **Define the Sample Utterances.** These are all the different phrases users can say to interact with your skill.

   Copy and paste the contents of the speechAssets/SampleUtterances.txt file included.


Sample Utterances
These are what people say to interact with your skill. Type or paste in all the ways that people can invoke the intents. Learn more

Up to 3 of these will be used as Example Phrases, which are hints to users.

```
GetNewFactIntent a fact
GetNewFactIntent a space fact
GetNewFactIntent tell me a fact
GetNewFactIntent tell me a space fact
GetNewFactIntent give me a fact
GetNewFactIntent give me a space fact
GetNewFactIntent tell me trivia
GetNewFactIntent tell me a space trivia
GetNewFactIntent give me trivia
GetNewFactIntent give me a space trivia
GetNewFactIntent give me some information
```

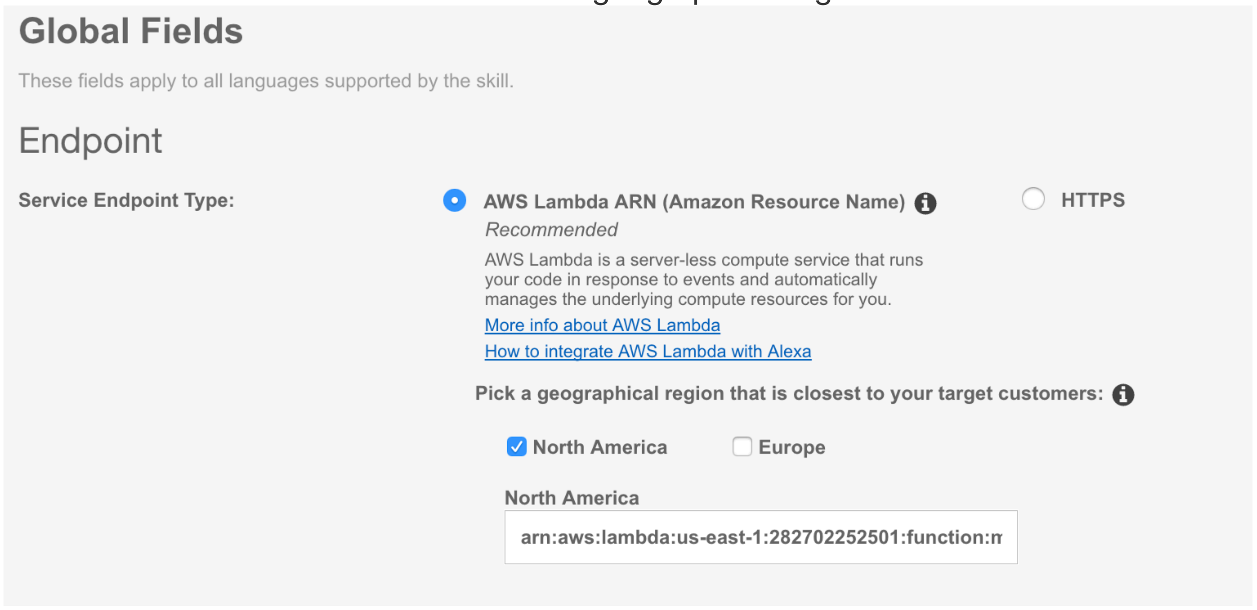
10. **Click the Save button to save the changes made to the Interaction Model.** It may take a couple of minutes for the model to save.

11. **Once the model has been saved, click the Next button to move on to Configuration.**

12. **Configure the Endpoint.** This is where we tell Alexa where to find the code logic to execute when a user utters a phrase corresponding to this skill.

    In our case, all our code logic is on an AWS Lambda instance, so we want to provide its ARN (Amazon Resource Name). This is the value we had copied in the final step of Part I of this lab. You can find it again by returning to your Amazon Develop Console and navigating to the corresponding skill.

    Be sure to select "North America" as the geographical region.



Global Fields

These fields apply to all languages supported by the skill.

Endpoint

Service Endpoint Type: ● AWS Lambda ARN (Amazon Resource Name) ○ HTTPS

Recommended

AWS Lambda is a server-less compute service that runs your code in response to events and automatically manages the underlying compute resources for you.

More info about AWS Lambda

How to integrate AWS Lambda with Alexa

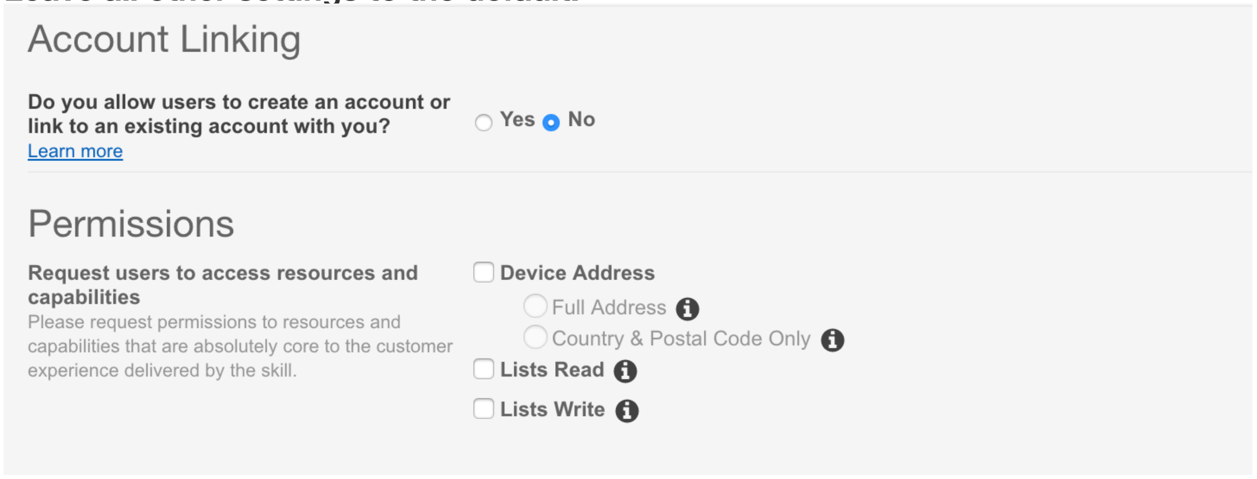
Pick a geographical region that is closest to your target customers:

[✓ North America] Europe

North America

arn:aws:lambda:us-east-1:282702252501:function:m

13. **Leave all other settings to the default.**



Account Linking

Do you allow users to create an account or link to an existing account with you? Learn more
Yes [✓ No]

Permissions

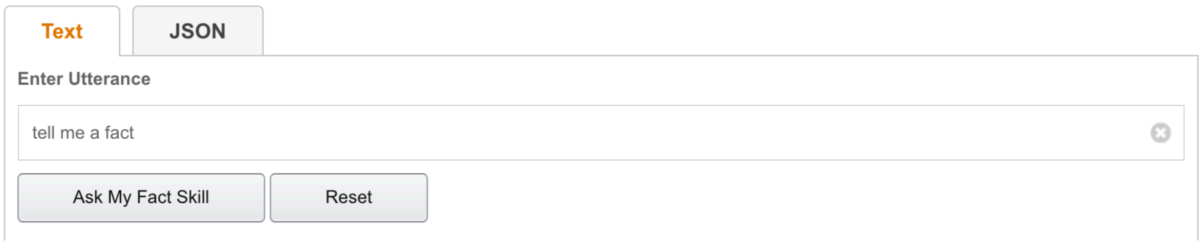
Request users to access resources and capabilities
Please request permissions to resources and capabilities that are absolutely core to the customer experience delivered by the skill.

- ☐ Device Address
  - ○ Full Address
  - ○ Country & Postal Code Only
- ☐ Lists Read
- ☐ Lists Write

14. **Click Save button to save the changes.**

15. **Click Next to move on to the final step of Testing.**

16. **You are now ready to start testing your sample skill!** In order to test it, you can try typing in some of the Sample Utterances from your speechAssets/SampleUtterances.txt file in the Service Simulator, and see how Alexa responds.



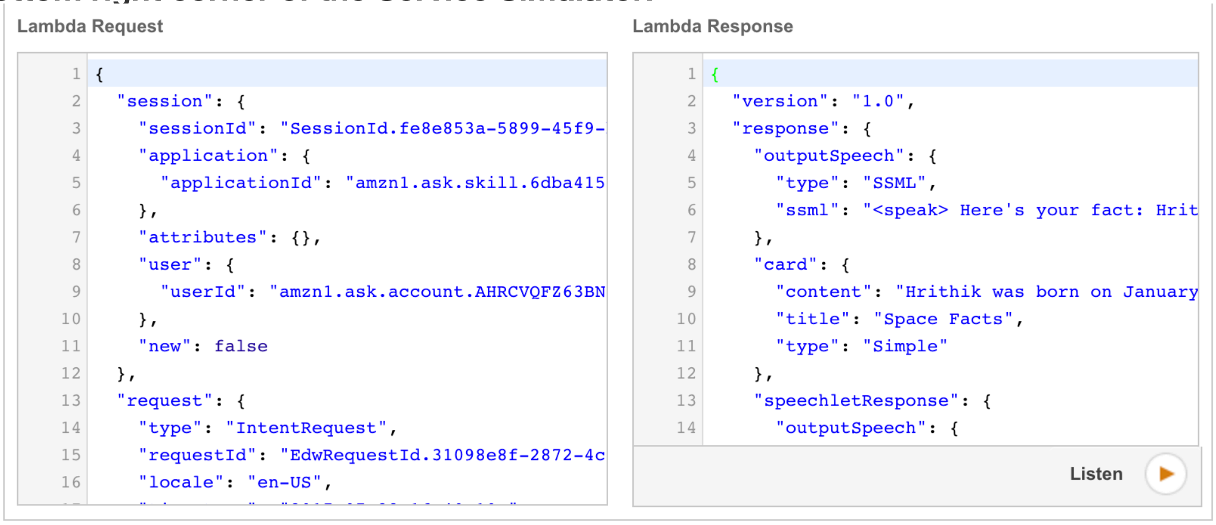
Text | JSON

Enter Utterance

tell me a fact

Ask My Fact Skill | Reset

17. **You can also test how Alexa responds using the “Listen” button in the bottom right corner of the Service Simulator.**



```
{
  "session": {
    "sessionId": "SessionId.fe8e853a-5899-45f9-
    "application": {
      "applicationId": "amzn1.ask.skill.6dba415
    },
    "attributes": {},
    "user": {
      "userId": "amzn1.ask.account.AHRCVQFZ63BN
    },
    "new": false
  },
  "request": {
    "type": "IntentRequest",
    "requestId": "EdwRequestId.31098e8f-2872-4c
    "locale": "en-US",
```

```
{
  "version": "1.0",
  "response": {
    "outputSpeech": {
      "type": "SSML",
      "ssml": "<speak> Here's your fact: Hrit
    },
    "card": {
      "content": "Hrithik was born on January
      "title": "Space Facts",
      "type": "Simple"
    },
    "speechletResponse": {
      "outputSpeech": {
```

Congratulations!

You just built your first Alexa Skill from scratch! You now know how to:

- Use the Amazon Developer Portal to configure the Voice User Interface of an Alexa Skill
- Use the AWS Lambda Service to develop the back end of an Alexa Skill
- Use the Alexa Service Simulator to test your Alexa Skill