Programming Alexa Skills

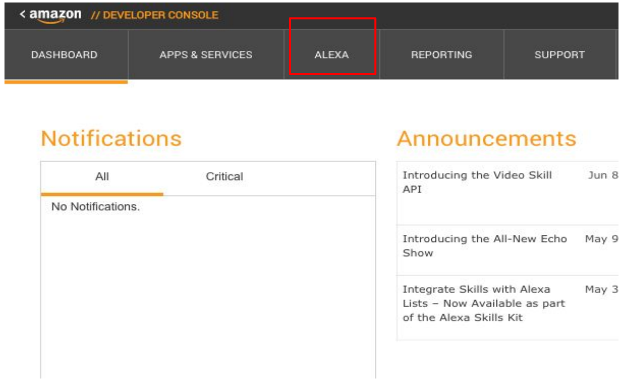
Steps

- Create Alexa Skill through Developer Portal
  - Sample Utterances
  - Intent Schema
  - Invocation names
- Create Lambda Function for serverless-hosting
  - Testing
  - Configuring state

Code located at https://github.com/hermano360/alexa-app-demo

Alexa Skill Configuration - Front End

1. Go to https://developer.amazon.com, sign in, select Alexa, Select Alexa Skills Kit, and Add New Skill



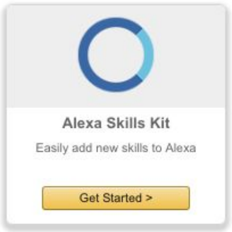
Notifications

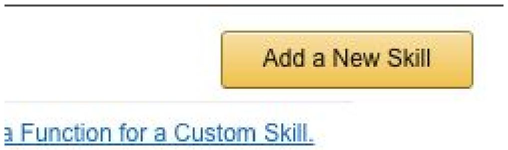
All | Critical

No Notifications.

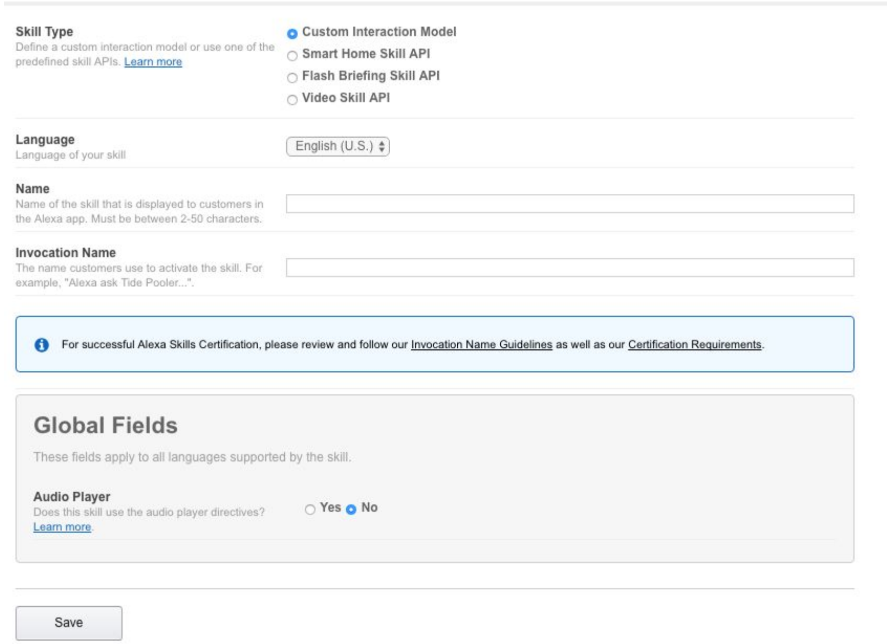
Announcements

Introducing the Video Skill API	Jun 8
Introducing the All-New Echo Show	May 9
Integrate Skills with Alexa Lists – Now Available as part of the Alexa Skills Kit	May 3





2. Choose Custom Interaction Model, Select Name for skill, Invocation name is how you will activate it



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 Video 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...".	

For successful Alexa Skills Certification, please review and follow our Invocation Name Guidelines as well as our Certification Requirements.

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]

Save

Intents, Slot Types, Sample Utterances

Intents - Essentially functions that allows Alexa to respond to user feedback.

Intent Schema - Allows user to configure the functions that will be available, in addition to specifying custom key slots (variables) that function may be expecting expecting

Slot Types - Gives Alexa preferences on certain expected words that will improve understanding.

Sample Utterances - Allows user to define user responses that will correspond to functions

Intent Schema, Custom Slot Types, Sample Utterances

Sample Utterances extending default Alexa Intents

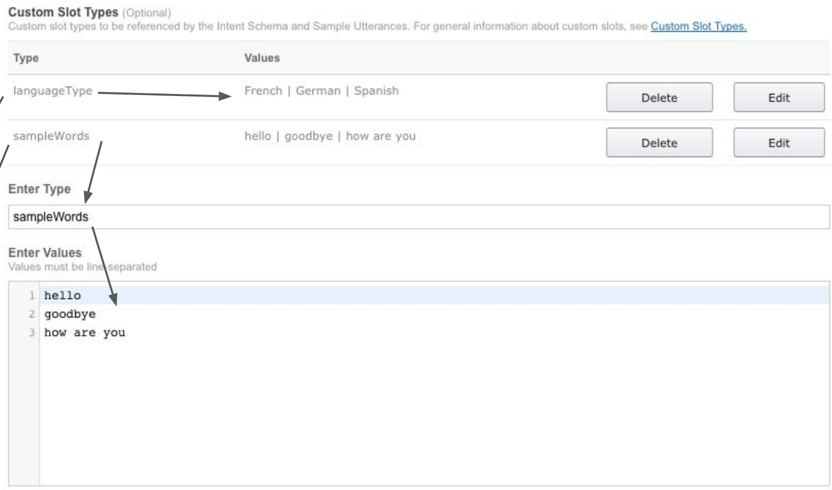
Alexa Intents

```
AMAZON.StopIntent stop
AMAZON.StopIntent off
AMAZON.StopIntent shut up
AMAZON.StopIntent stop
AMAZON.StopIntent off
AMAZON.StopIntent shut up
AMAZON.StartOverIntent start over
AMAZON.StartOverIntent restart
AMAZON.StartOverIntent start again
AMAZON.StartOverIntent play again
AMAZON.StartOverIntent start over
AMAZON.StartOverIntent restart
AMAZON.StartOverIntent start again
AMAZON.StartOverIntent play again
LanguageIntent I want {Language}
LanguageIntent How About {Language}
LanguageIntent Translate to {Language}
LanguageIntent {Language}
TranslationIntent translate {TranslationWord}
TranslationIntent what does {TranslationWord} mean
TranslationIntent do {TranslationWord}
```

Sample Utterances for Custom Intent

```
{
 "intents": [
  {
   "intent": "AMAZON.YesIntent"
  },
  {
   "intent": "AMAZON.NoIntent"
  },
  {
   "intent": "AMAZON.HelpIntent"
  },
  {
   "intent": "AMAZON.StopIntent"
  },
  {
   "intent": "AMAZON.CancelIntent"
  },
  {
   "intent": "AMAZON.StartOverIntent"
  },
  {
   "slots": [
    {
     "name": "Language",
     "type": "languageType"
    }
   ],
   "intent": "LanguageIntent"
  },{
   "slots": [
    {
     "name": "TranslationWord",
     "type": "sampleWords"
    }
   ],
   "intent": "TranslationIntent"
  }
 ]
}
```

Custom Intents



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.

Type	Values
languageType	French | German | Spanish
sampleWords	hello | goodbye | how are you

Enter Type
sampleWords

Enter Values
Values must be line-separated

```
hello
goodbye
how are you
```

Lambda Function Basics - Alexa Skills API

```
this.emit(':ask', welcomeMessage, repeatWelcomeMessage);
```

```
this.emit(':tell', welcomeMessage, repeatWelcomeMessage);
```

Lambda Function Basics

Library that allows access to Alexa api is alexa-sdk

Define the ‘states’ that the app will find itself to allow for more complex and conditional interactions

```
var Alexa = require('alexa-sdk');

var states = {
    STARTMODE: '_STARTMODE',                    // Alexa is prompting the user to start or restart the interaction in this state
    ASKLANGUAGE: '_ASKLANGUAGE',                // Alexa is asking user the questions in this state
    ASK_FRENCH_WORDS: '_ASK_FRENCH_WORDS',      // Alexa is asking the words to be translated in French in this state
    ASK_SPANISH_WORDS: '_ASK_SPANISH_WORDS',    // Alexa is asking the words to be translated in Spanish this state
    ASK_GERMAN_WORDS: '_ASK_GERMAN_WORDS',      // Alexa is asking the words to be translated in German this state
    TRANSLATION_GIVEN: '_TRANSLATION_GIVEN'     // Alexa is giving translating, waiting to restart
};
```

Lambda Function Basics

Defining how Alexa will Interact with user

```
// These are messages that Alexa says to the user during conversation

// This is the initial welcome message
var welcomeMessage = "Welcome to Translate This. Would you like to continue?";

// This is the message that is repeated if the response to the initial welcome message is not heard
var repeatWelcomeMessage = "Would you like to translate some words?";

// this is the message that is repeated if Alexa does not hear/understand the reponse to the welcome message
var promptToStartMessage = "Please say yes to continue or no to quit. ";

// This is the prompt during the app when Alexa doesnt hear or understand a yes / no reply
var promptToSayYesNo = "Please repeat your yes or no answer";

// This is the prompt for language
var promptForLanguage = "Please indicate which language you would like, Spanish, French, or German?";

// This is the reprompt for language
var repromptForLanguage = "Please say Spanish, French, or German";

// This is the prompt in case of incorrect language
var incorrectLanguage = "That language is not yet supported. Please say Spanish, French, or German.";
```

Lambda Function Basics

Defining how Alexa will Interact with user

```
// This is the prompt in case of incorrect language
var incorrectLanguage = "That language is not yet supported. Please say Spanish, French, or German.";

// Prompt for word
var promptForWord = "Please choose one of the following phrases. Hello. Goodbye. How are you.";

// reprompt for word
var repromptForWord = "Please say Hello, Goodbye, or How are you";

// responding to unsupported word said
var incorrectWord = "That word or phrase is not supported. Please say Hello, Goodbye, or How are you";


// this is the help message during the setup at the beginning of the game
var helpMessage = `Translation This App will help you to say a couple key phrases in Spanish, French, and German.
                The app will prompt you for a language and then you can choose a phrase to translate.
                Would you like to continue?`;

// this is the repate help message during the setup at the beginning of the game
var helpMessageRepeat = "Would you like to continue?";

// This is the goodbye message when the user has asked to quit the game
var goodbyeMessage = "Until next time!";
```

Lambda Function Basics

```
// This is the simulated translation object. To be replaced with Translation API
var translations = {
    fr:{
        hello: 'bone jur',
        goodbye: 'oar eh vooah',
        'how are you': 'como taleh vu'
    },
    sp:{
        hello: 'oh la',
        'goodbye': 'ah di owes',
        'how are you': 'como estas'
    },
    de:{
        hello: 'hallo',
        goodbye: 'off veeder sen',
        'how are you': 'vee get es dear'
    }
};
```

The translation object with sample translations and phonetic mock responses. A combination of services such as Google Translate and Amazon Polly can be used to provide the correct translation and a pronounciation in the native tongue.

Lambda Function Basics

Registering the handlers that will be available to Alexa

```
// Called when the session starts.
exports.handler = function (event, context, callback) {
  var alexa = Alexa.handler(event, context);
  alexa.registerHandlers(newSessionHandler, startTranslationHandlers, askLanguageHandlers, frenchHandlers, spanishHandlers, germanHandlers, translateHandlers);
    alexa.execute();
};
```

Defines all of the handlers that the app will register and respond to

Lambda Function Basics

```
// set state to start up and  welcome the user
var newSessionHandler = {
    'NewSession': function () {
        this.handler.state = states.STARTMODE;
        this.emit(':ask', welcomeMessage, repeatWelcomeMessage);
    }
};
```

Alexa will look for this handler as the default start handler

Allowing you to tap into the STARTMODE handler

These two messages are what is going to prompt the user. The responses will be handled by the STARTMODE handler

Lambda Function Basics

```
// Once the user has agreed to continue, this is how the program will ask for language.
var startTranslationHandlers = Alexa.CreateStateHandler(states.STARTMODE, {
    'AMAZON.YesIntent': function () {
        // set state to asking for Language
        this.handler.state = states.ASKLANGUAGE;
        // ask the first question
        this.emit(':ask', promptForLanguage, repromptForLanguage);
    },
    'AMAZON.NoIntent': function () {
        // Handle No intent.
        this.emit(':tell', goodbyeMessage);
    },
    'AMAZON.StopIntent': function () {
        this.emit(':tell', goodbyeMessage);
    },
    'AMAZON.CancelIntent': function () {
        this.emit(':tell', goodbyeMessage);
    },
    'AMAZON.StartOverIntent': function () {
         this.emit(':ask', promptToStartMessage, promptToStartMessage);
    },
    'AMAZON.HelpIntent': function () {
        this.emit(':ask', helpMessage, helpMessageRepeat);
    },
    'Unhandled': function () {
        this.emit(':ask', promptToStartMessage, promptToStartMessage);
    }
});
```

This assigns this function as the handler for the STARTMODE state

If the user agreed to continue in the previous step, the program now assigns the state to ASKLANGUAGE, which is responsible to handling a language. We are prompting the user for the languages

Lambda Function Basics

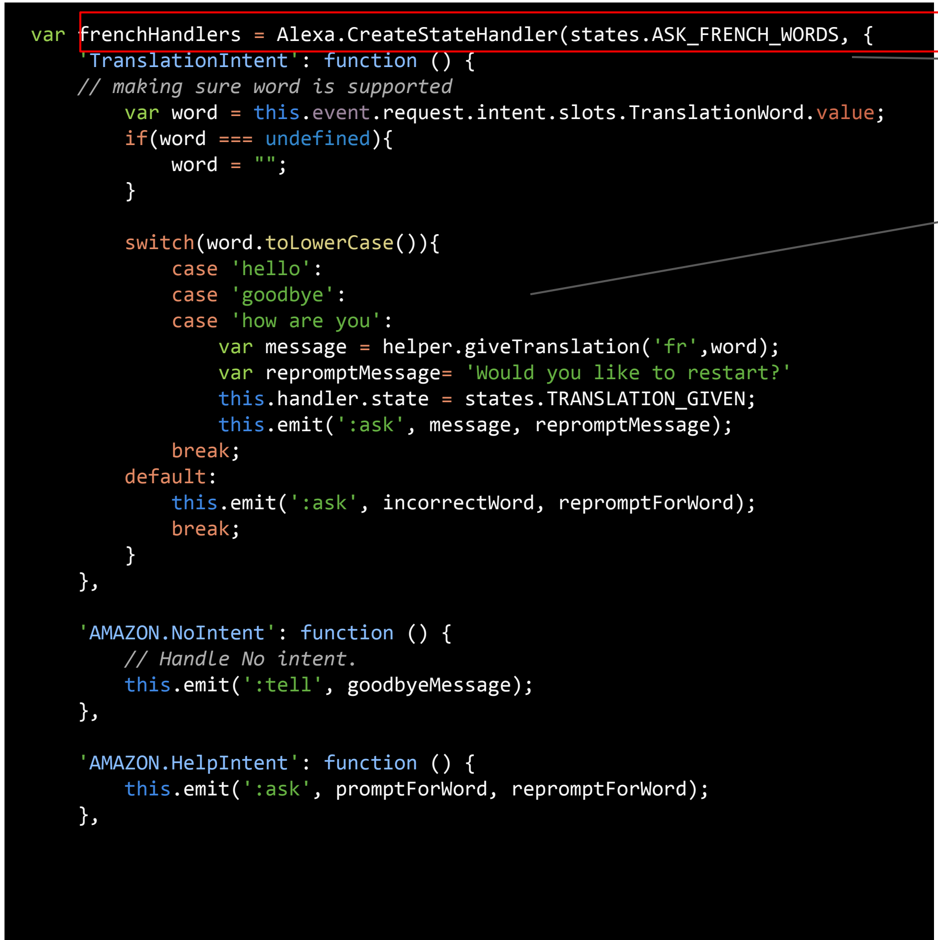
```
var askLanguageHandlers = Alexa.CreateStateHandler(states.ASKLANGUAGE, {
    'LanguageIntent': function () {
    // making sure language is supported
    var language = this.event.request.intent.slots.Language.value;
    if(language === undefined){
        language = "";
    }
    switch(language.toLowerCase()){
        case 'spanish':
            this.handler.state = states.ASK_SPANISH_WORDS;
            this.emit(':ask', promptForWord, repromptForWord);
            break;
        case 'german':
            this.handler.state = states.ASK_GERMAN_WORDS;
            this.emit(':ask', promptForWord, repromptForWord);
            break;
        case 'french':
            this.handler.state = states.ASK_FRENCH_WORDS;
            this.emit(':ask', promptForWord, repromptForWord);
            break;
        default:
            this.emit(':ask', incorrectLanguage, repromptForLanguage);
    }
},
    'AMAZON.HelpIntent': function () {
        this.emit(':ask', promptForLanguage, repromptForLanguage);
    },
```

This is the handler for the ASKLANGUAGE state.

If the LanguageIntent is triggered by one of the sample utterances mentioned previously, the slot value is registered and analyzed. The appropriate state will be assigned (i.e ASK_SPANISH_WORDS). The program will now prompt for words.

```
    'AMAZON.StopIntent': function () {
        this.emit(':tell', goodbyeMessage);
    },
    'AMAZON.CancelIntent': function () {
        this.emit(':tell', goodbyeMessage);
    },
    'AMAZON.StartOverIntent': function () {
        // reset the game state to start mode
        this.handler.state = states.STARTMODE;
        this.emit(':ask', welcomeMessage, repeatWelcomeMessage);
    },
    'Unhandled': function () {
        this.emit(':ask', promptForLanguage, repromptForLanguage);
    }
});
```

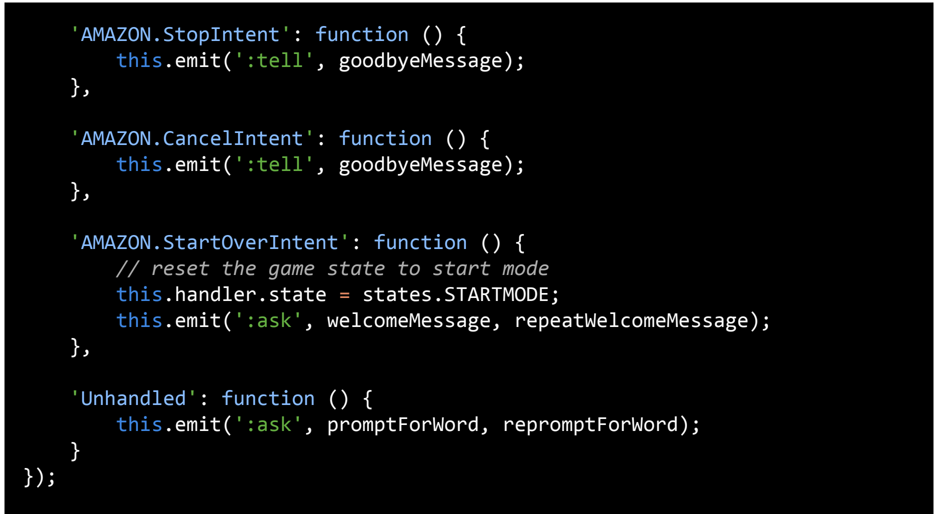
Lambda Function Basics



```
var frenchHandlers = Alexa.CreateStateHandler(states.ASK_FRENCH_WORDS, {
    'TranslationIntent': function () {
    // making sure word is supported
        var word = this.event.request.intent.slots.TranslationWord.value;
        if(word === undefined){
            word = "";
        }

        switch(word.toLowerCase()){
            case 'hello':
            case 'goodbye':
            case 'how are you':
                var message = helper.giveTranslation('fr',word);
                var repromptMessage= 'Would you like to restart?'
                this.handler.state = states.TRANSLATION_GIVEN;
                this.emit(':ask', message, repromptMessage);
            break;
        default:
            this.emit(':ask', incorrectWord, repromptForWord);
            break;
        }
    },

    'AMAZON.NoIntent': function () {
        // Handle No intent.
        this.emit(':tell', goodbyeMessage);
    },

    'AMAZON.HelpIntent': function () {
        this.emit(':ask', promptForWord, repromptForWord);
    },
```

This is the handler for the ASK_FRENCH_WORDS state.

If the translation intent is triggered by the previous user response, the words are logged. If it is one of the supported words, the helper function is called to give the translation. If not, the user is reprompted



```
    'AMAZON.StopIntent': function () {
        this.emit(':tell', goodbyeMessage);
    },

    'AMAZON.CancelIntent': function () {
        this.emit(':tell', goodbyeMessage);
    },

    'AMAZON.StartOverIntent': function () {
        // reset the game state to start mode
        this.handler.state = states.STARTMODE;
        this.emit(':ask', welcomeMessage, repeatWelcomeMessage);
    },

    'Unhandled': function () {
        this.emit(':ask', promptForWord, repromptForWord);
    }
});
```

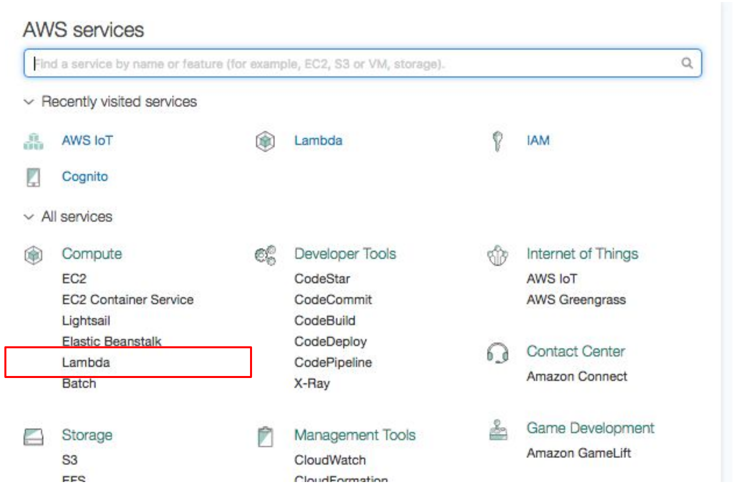
Lambda Function Basics

```
var helper = {
    giveTranslation: function (language, word) {
        console.log(language,word)
        var translatedWord = translations[language][word];
        var message = `The translation for ${word} is ${translatedWord}, would you like to restart?`;
        return message


    }
}
```

This helper object is a convenient place to put methods that can be used throughout the app. This method gets the language of interest, indicated by the state that the app is in as well as the word and gives the translation based on the translation object previously shown. The message is returned for use in the app

AWS Lambda Function

1. Enter Lambda main screen and ensure you are either in N.Virginia or Ireland



AWS services

Find a service by name or feature (for example, EC2, S3 or VM, storage).

Recently visited services

AWS IoT | Lambda | IAM
Cognito

All services

Compute: EC2, EC2 Container Service, Lightsail, Elastic Beanstalk, Lambda, Batch

Developer Tools: CodeStar, CodeCommit, CodeBuild, CodeDeploy, CodePipeline, X-Ray

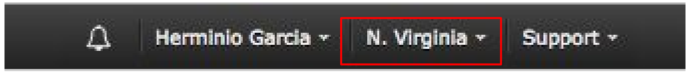
Internet of Things: AWS IoT, AWS Greengrass

Contact Center: Amazon Connect

Storage: S3, EFS

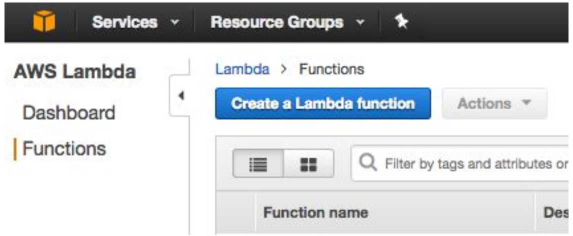
Management Tools: CloudWatch, CloudFormation

Game Development: Amazon GameLift



Herminio Garcia ▾ | N. Virginia ▾ | Support ▾

2. Create a Lambda Function and select Blank Function, Select Alexa Skills Kit



Services ▾ | Resource Groups ▾

AWS Lambda

Dashboard

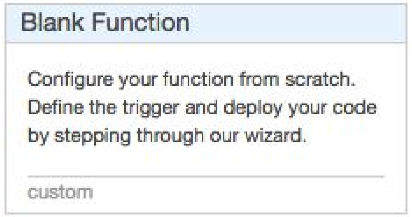
Functions

Lambda > Functions

Create a Lambda function | Actions ▾

Filter by tags and attributes or

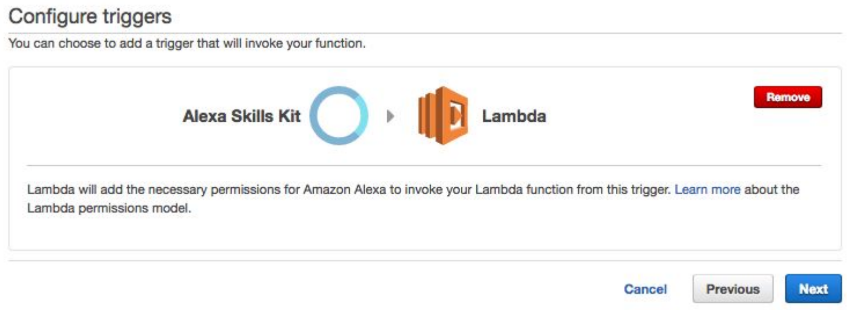
Function name | Des



Blank Function

Configure your function from scratch. Define the trigger and deploy your code by stepping through our wizard.

custom



Configure triggers

You can choose to add a trigger that will invoke your function.

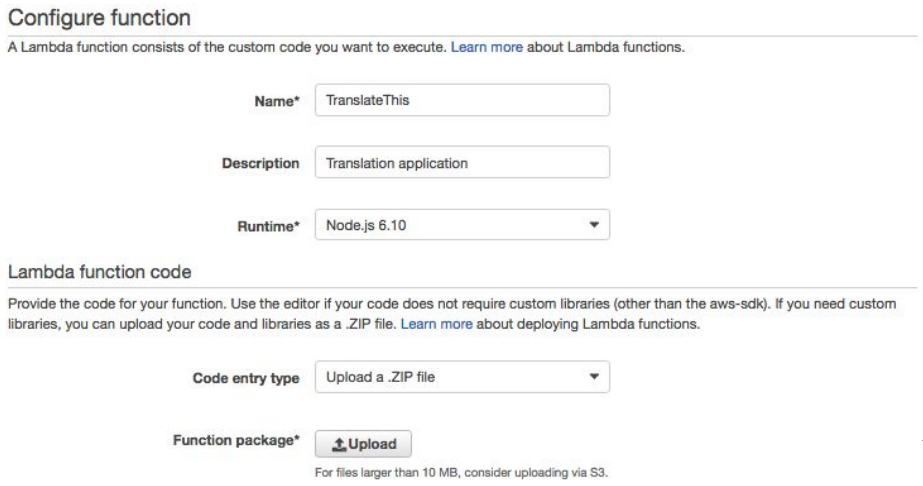
Alexa Skills Kit → Lambda | Remove

Lambda will add the necessary permissions for Amazon Alexa to invoke your Lambda function from this trigger. Learn more about the Lambda permissions model.

Cancel | Previous | Next

Lambda Function Basics - Uploading

Once Lambda function is complete, it will be uploaded to the Lambda console.


Configure function

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

Name*	TranslateThis
Description	Translation application
Runtime*	Node.js 6.10

Lambda function code

Provide the code for your function. Use the editor if your code does not require custom libraries (other than the aws-sdk). If you need custom libraries, you can upload your code and libraries as a .ZIP file. Learn more about deploying Lambda functions.

Code entry type	Upload a .ZIP file
Function package*	Upload

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

In order to create the zip file, have the code you wrote in a file named index.js.

You will need a package.json file like this:

```
{
  "name": "Translatethis",
  "version": "1.0.0",
  "description": "Gives you basic important translations in various languages",
  "main": "index.js",
  "dependencies": {
    "alexa-sdk": "^1.0.0"
  },
  "devDependencies": {},
  "author": "Herminio Garcia",
  "license": "ISC"
}
```

You run: npm install to create the node_modules folder.

Compress the 3 files: node_modules, index.js, and package.json. Upload the resulting file

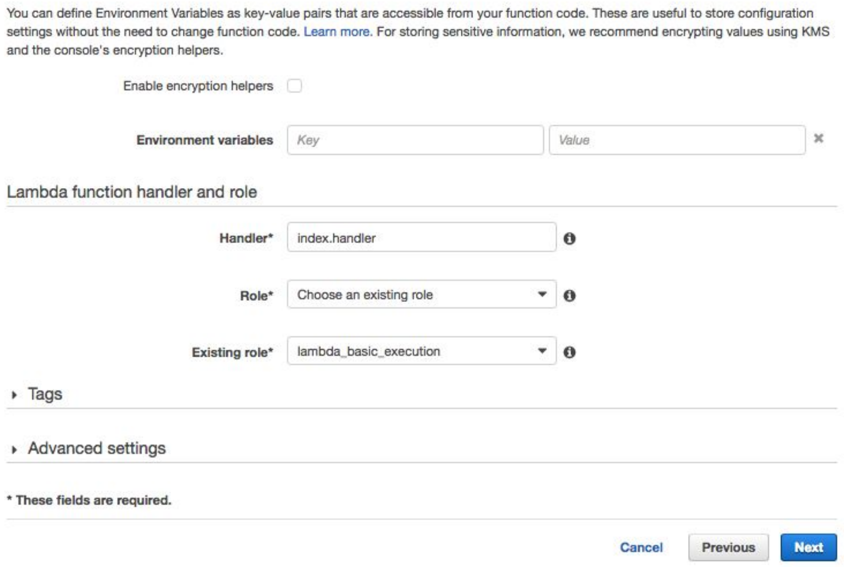
Lamda

For Handler click on: index.handler

For Role: Choose and Existing Role

Existing Role: lambda_basic_execution

Click on Next



You can define Environment Variables as key-value pairs that are accessible from your function code. These are useful to store configuration settings without the need to change function code. Learn more. For storing sensitive information, we recommend encrypting values using KMS and the console's encryption helpers.

Enable encryption helpers ☐

Environment variables Key Value

Lambda function handler and role

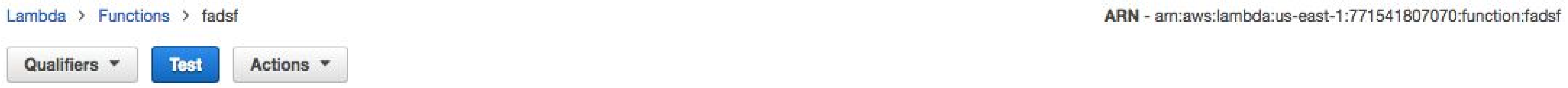
Handler* index.handler

Role* Choose an existing role

Existing role* lambda_basic_execution

*** These fields are required.**

Lambda Upload


Lambda › Functions › fadsf

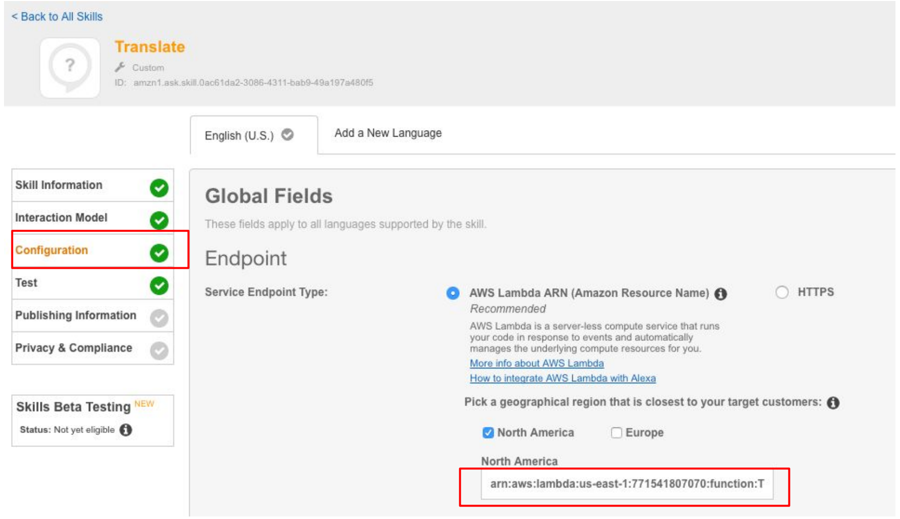
ARN - arn:aws:lambda:us-east-1:771541807070:function:fadsf

To configure tests, click on Actions: Configure Test Event: Sample Event Template: Alexa Start Session

Save the ARN number, it will be necessary for the Alexa Skill Configuration

Alexa Skill Configuration - Front End

1. Go back to https://developer.amazon.com, and click on your application. Go to Configuration on the side menu. Click on AWS Lambda, North America or Europe, and type on the obtained ARN number



< Back to All Skills

Translate

Custom

ID: amzn1.ask.skill.0ac61da2-3086-4311-bab9-49a197a480f5

English (U.S.) ✓ | Add a New Language

- Skill Information ✓
- Interaction Model ✓
- Configuration ✓
- Test ✓
- Publishing Information
- Privacy & Compliance

Skills Beta Testing NEW

Status: Not yet eligible

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-1:771541807070:function:T

Click Next. Now you can test the application. Go to Service simulator. Type in ‘Alexa, open’ and the name of your application. I.e. Alexa open Translate This. The response will be shown in the lambda response and you can now type in the responses as you would a normal conversation.

You now have a working Alexa App