

ORACLE®

Oracle Digital Assistant

The Complete Training

Webhook

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Topic agenda

- 1 ➤ Overview
- 2 ➤ Creating webhook clients with Node.js
- 3 ➤ Voice integration with Alexa

Topic agenda

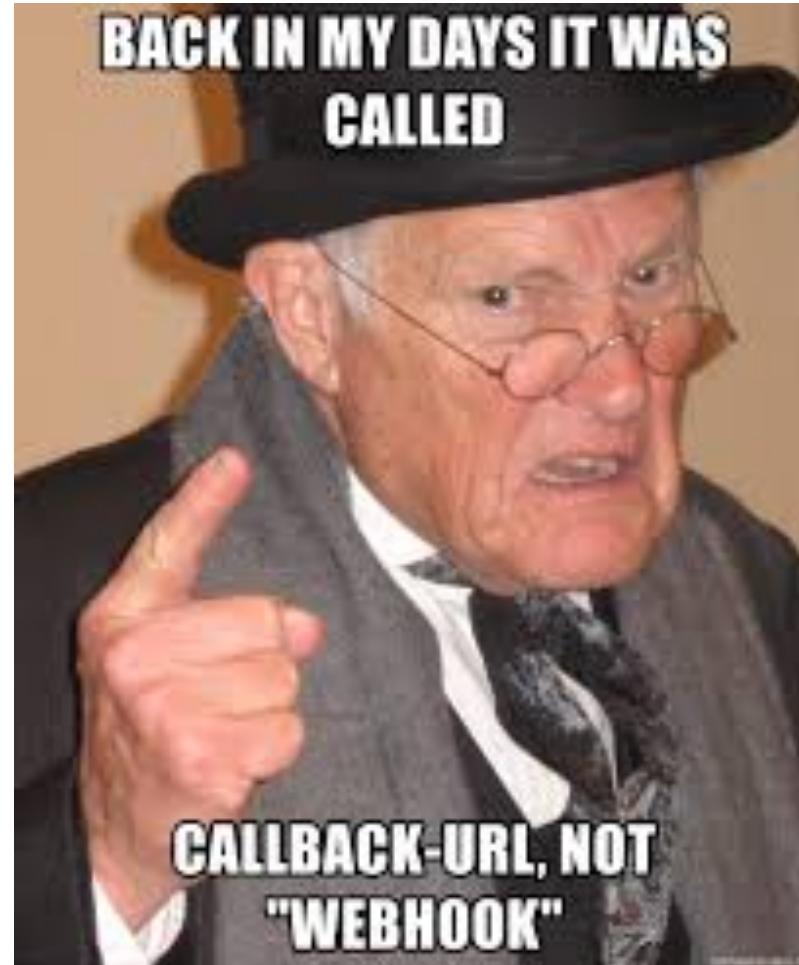
1 ➤ Overview

2 ➤ Creating webhook clients with Node.js

3 ➤ Voice integration with Alexa

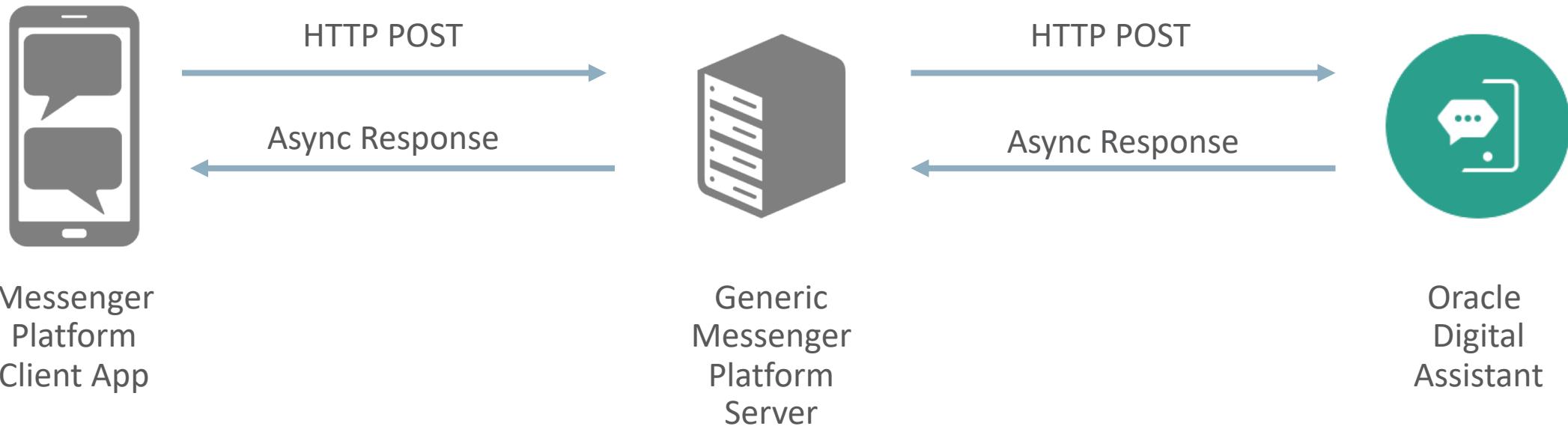


What is a webhook?



*<https://memegenerator.net/>

Generic HTTP webhook channel



Generic HTTP webhook support

- Generic channel mechanism
 - Supporting other messenger platforms
 - SMS, Alexa, Slack, etc
 - Virtually any messaging platform that supports HTTP Webhooks
- Through the webhook channel
 - Bot channel publishes an HTTP Endpoint to receive messages
 - You define a response HTTP Endpoint
 - Bot will send response messages back to your server
- To verify messages the Intelligent Bot publishes a secret key in the Webhook channel
- Caller must supply
 - X-Hub-Signature HTTP header
 - Set to SHA256 signature of payload
 - Secret key used as SHA key
- Uses same mechanism on response
 - Optional for caller to verify payload

Topic agenda

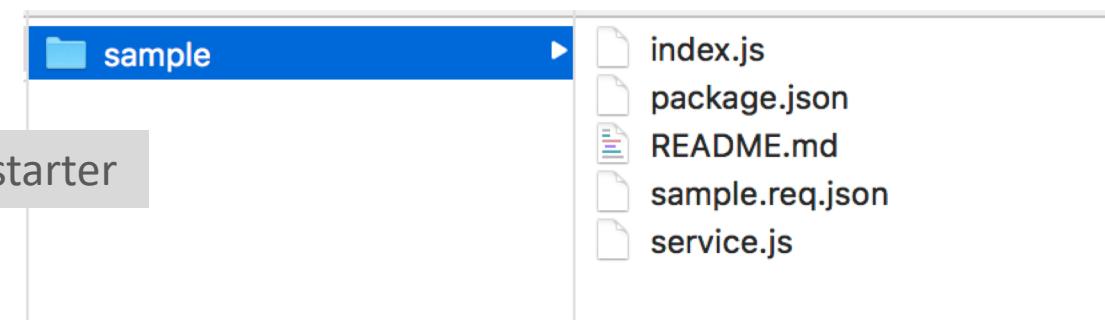
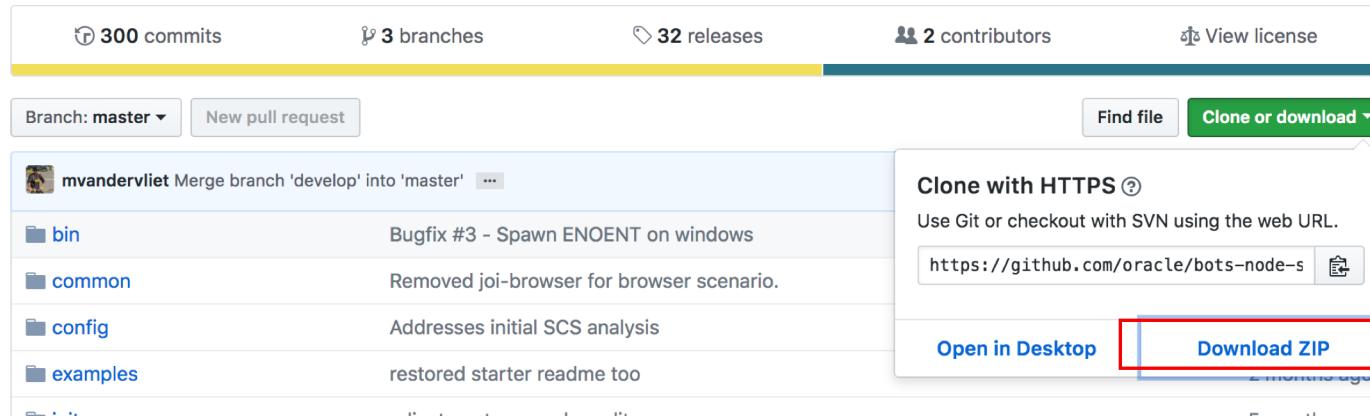
1 Overview

2 Creating webhook clients with Node.js

3 Voice integration with Alexa

Create webhook client

Oracle Bots Node.js SDK <https://oracle.github.io/bots-node-sdk/>



Navigate to bots-node-sdk-master/examples/Webhook/starter

Create webhook client

The screenshot shows a Mac OS X desktop environment. On the left, a terminal window titled "sample — bash — 80x24" displays the command-line process of creating a webhook client. On the right, a file browser window titled "Today" shows the directory structure of the project.

Terminal Output:

```
[rdhamija-mac:sample rohitdhamija$ pwd  
/Users/rohitdhamija/Documents/RDHWorkSpace/MOBILECLOUDTEAM/PROJECTS/webhook/sample  
[rdhamija-mac:sample rohitdhamija$ npm install  
npm notice created a lockfile as package-lock.json. You should commit this file.  
npm WARN oracle-bot-webhook@1.0.0 No repository field.  
npm WARN oracle-bot-webhook@1.0.0 No license field.  
  
added 60 packages in 18.005s  
[rdhamija-mac:sample rohitdhamija$ npm install --save @oracle/bots-node-sdk  
npm WARN oracle-bot-webhook@1.0.0 No repository field.  
npm WARN oracle-bot-webhook@1.0.0 No license field.  
  
+ @oracle/bots-node-sdk@2.1.3  
updated 1 package in 0.846s  
rdhamija-mac:sample rohitdhamija$
```

File Browser:

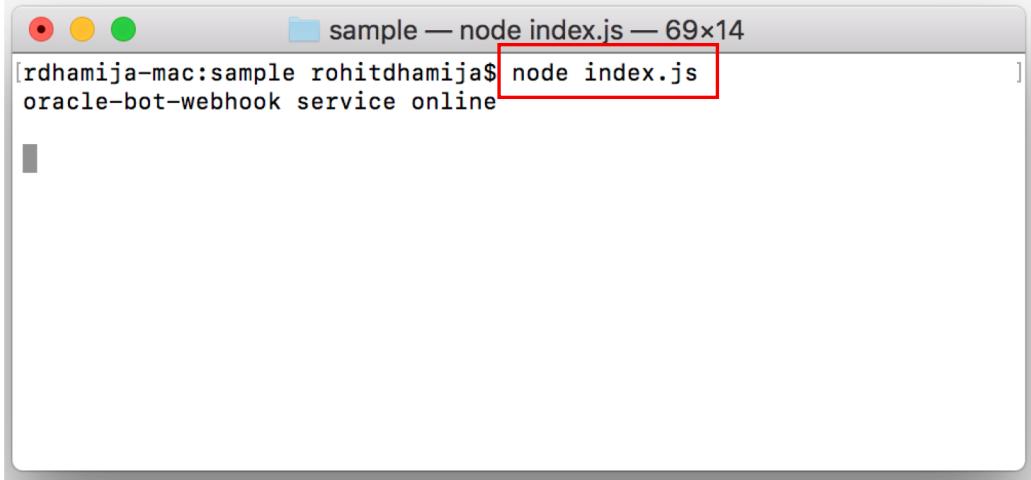
Today	Today
sample	node_modules
package-lock.json	
package.json	
2018	
index.js	
README.md	
sample.req.json	
service.js	

Configuration

```
19     webhook
20         .on(WebhookEvent.ERROR, err => logger.error('Error:', err.message))
21         .on(WebhookEvent.MESSAGE_SENT, message => logger.info('Message to bot:', message))
22     ▼   .on(WebhookEvent.MESSAGE_RECEIVED, message => {
23         // message was received from bot. forward to messaging client.
24         logger.info('Message from bot:', message);
25         // TODO: implement send to client...
26     });
27
28     // Create endpoint for bot webhook channel configuration (Outgoing URI)
29     // NOTE: webhook.receiver also supports using a callback as a replacement for WebhookEvent.MESSAGE_RECEIVED.
30     // - Useful in cases where custom validations, etc need to be performed.
31     app.post('/bot/message', webhook.receiver());
32
33     // Integrate with messaging client according to their specific SDKs, etc.
34     ▼ app.post('/test/message', (req, res) => {
35         const { user, text } = req.body;
36         // construct message to bot from the client message format
37         const MessageModel = webhook.MessageModel();
38     ▼   const message = {
39         userId: user,
40         messagePayload: MessageModel.textConversationMessage(text)
41     };
42         // send to bot webhook channel
43         webhook.send(message)
44             .then(() => res.send('ok'), e => res.status(400).end(e.message));
45     });
46 }
```

Run your sample

```
1 const express = require('express');
2 const service = require('./service');
3 const pkg = require('./package.json');
4
5 const logger = console;
6 const app = express();
7 service(app);
8
9 ▼ const server = app.listen(process.env.PORT || 3000, () => {
10   logger.info(`[${pkg.name}] service online\n`);
11 });
12
13 module.exports = server;
14
```



A terminal window titled "sample — node index.js — 69x14". The command "node index.js" is highlighted with a red box. The output shows the application logging "oracle-bot-webhook service online".

```
[rdhamija-mac:sample rohitdhamija$ node index.js
oracle-bot-webhook service online]
```



Cannot GET /

Topic Agenda

- 1 ➔ Overview
- 2 ➔ Creating Webhook clients with Node.js (build, configure)
- 3 ➔ Voice integration with Alexa

Alexa Integration

Integration Architecture

Architecture elements

Amazon Alexa

Custom Alexa Skill



Web Server app

Node.js Alexa
skill code app
on ACCS

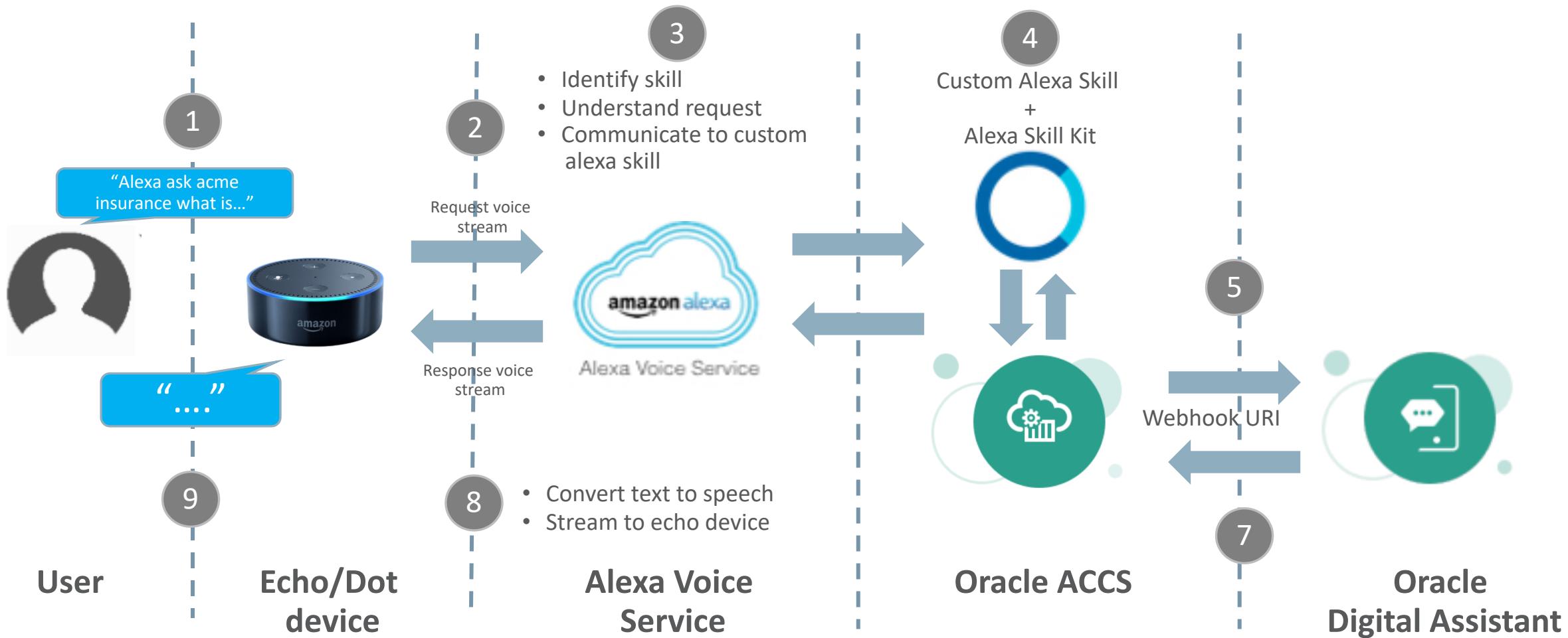


Oracle Digital
Assistant

Create
Webhook
channel



Alexa integration reference architecture



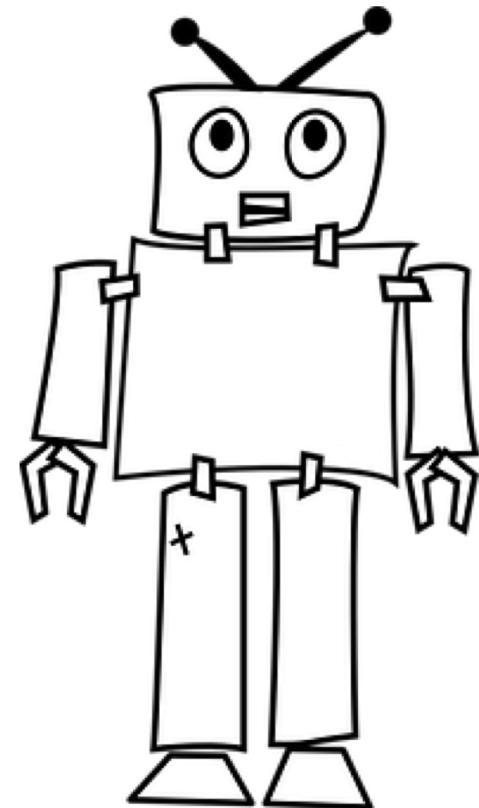
Alexa integration

Amazon Alexa

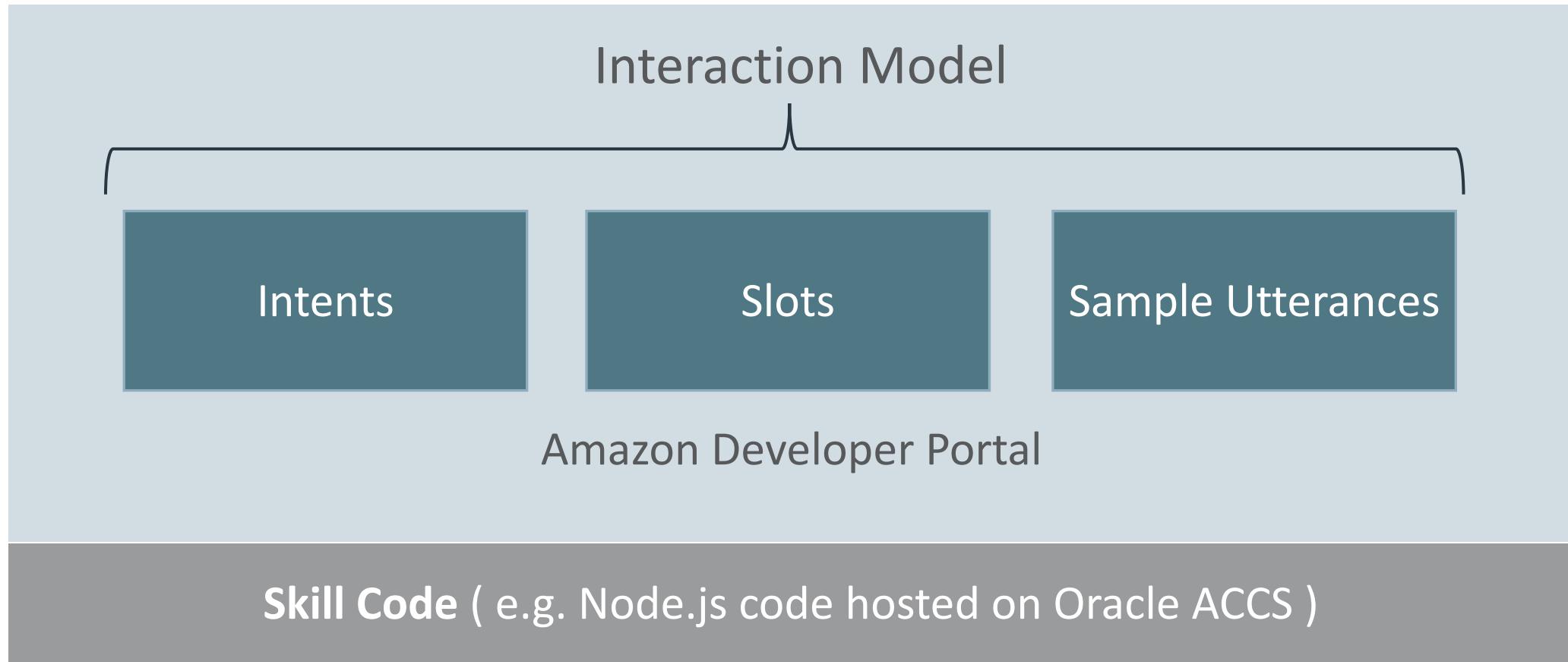
Key terminologies

- Skill
 - Capabilities built by third party to extend core skills of Alexa
 - Built-in skills: e.g. time, weather etc.
 - Third-party skills: Custom, smart home, flash briefing skills
- Alexa Voice Service (AVS)
 - Performs speech to text (and vice versa)
- Alexa Skill Kit (ASK)
 - Collection of API and tools to build new skills

You need to develop an **Alexa Skill** to
integrate Alexa as a channel with
Oracle Intelligent Bots

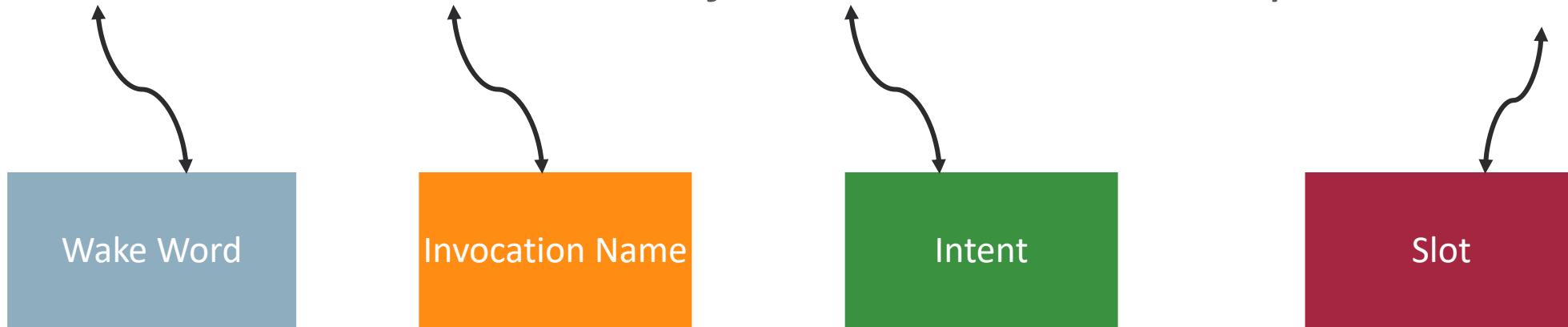


Alexa custom skill components



Parsing invocation name, intent and slots

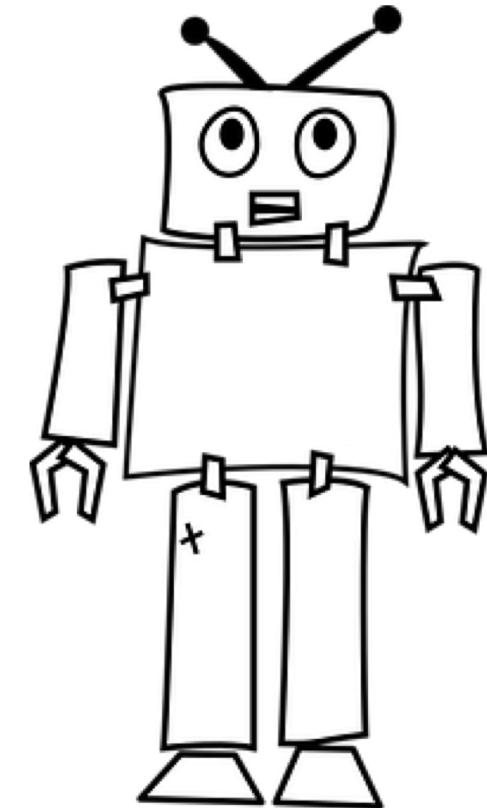
Alexa, ask **acme insurance** if **travel insurance** is required in **India**?



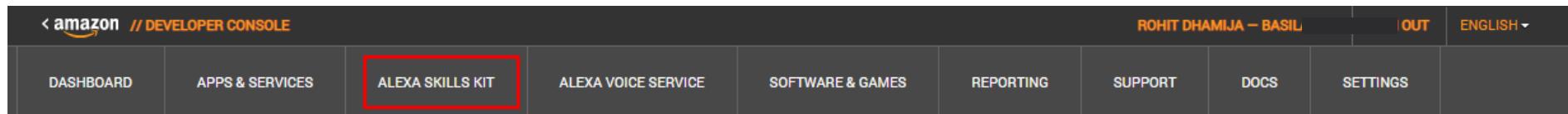
Alexa Integration

Develop Amazon Alexa Custom Skill

A valid **Amazon Developer account**
is needed to create Amazon skills



Amazon developer console



Notifications

All	Critical
No Notifications.	

Announcements

Alexa Skills Kit Expands to Mexico	Aug 8, 2018	Alexa Now Supports Kid Skills in India	Jul 2, 2018
Build Alexa Skills for Customers in Italy and Spain	Jun 18, 2018	Alexa and Echo Devices Now Available to Customers in France	Jun 5, 2018
Announcing New Way to Improve #AlexaSkill Discoverability and Enhance Customer Experience	May 30, 2018	Developer Preview: Easily Use Amazon Polly Voices in Alexa Skills	May 16, 2018

<https://developer.amazon.com/home.html>

Amazon skills kit developer console

Welcome to the Alexa Skills Kit Developer Console

Visit our [release notes](#) to learn about new feature and tools. Curious about what's new? [watch this video](#) or [read our documentation](#).

[Skills](#) [Earnings](#) [Payments](#)

Alexa Skills

[Create Skill](#)

SKILL NAME	LANGUAGE	TYPE	MODIFIED	STATUS	ACTIONS
 24hoursflowers <small>View Skill ID</small>	English (US)	Custom	2018-09-04	● In Development	Analytics Edit Delete

Creating a skill

Create a new skill

Skill name

acme insurance

14/50 characters

Default language

English (US)

More languages can be added to your skill after creation

Choose a model to add to your skill

There are many ways to start building a skill. You can design your own custom model or start with a pre-built model. Pre-built models are interaction models that contain a package of intents and utterances that you can add to your skill.

Custom SELECTED

Design a unique experience for your users. A custom model enables you to create all of your skill's interactions.

"Alexa, what's in the news?"

Flash Briefing

Give users control of their news feed. This pre-built model lets users control what updates they listen to.

"Alexa, turn on the kitchen lights"

Smart Home

Give users control of their smart home devices. This pre-built model lets users turn off the lights and other devices without getting up.

"Alexa, play Interstellar"

Video

Let users find and consume video content. This pre-built model supports content searches and content suggestions.

Create skill

Creating a skill – choose a template

alexia developer console

Your Skills acme insurance Build Test Distribution Certification Analytics

Choose a template

Select a quick start template to get started with a predefined skill or simply "Start from scratch"

Start from scratch SELECTED

Design a unique experience for your users and define your custom model from scratch.

 **Fact Skill**

Provides a list of interesting facts about a topic, Alexa will select a fact at random and tell it to the user when the skill is invoked. Includes 1 custom intent, and 4 built-in intents.

 **Quiz Game Skill**

Provides a list of interesting facts about a topic, Alexa will quiz a user with facts from the list. Includes 1 custom intent with 1 slot, and 6 built-in intents.

 **High-Low Game Skill**

Try to guess the target number. Alexa tells the player if the target number is higher or lower than their current guess. Includes 2 custom intents with 5 slots, and 5 built-in intents.

Get the skill ID

Skills	Earnings	Payments			
Alexa Skills					Create Skill
SKILL NAME	LANGUAGE	TYPE	MODIFIED	STATUS	ACTIONS
 acme insurance View Skill ID	English (US)	Custom	2018-09-04	In Development	Analytics Edit Delete
 24hoursflowers View Skill ID	English (US)	Custom	2018-09-04	In Development	Analytics Edit Delete

Get the skill ID

The screenshot shows the Alexa Skills page with two skills listed:

SKILL NAME	LANGUAGE	LAST UPDATED	STATUS	ACTIONS
acme insurance	English (US)	Custom	2018-09-04	In Development Analytics Edit Delete
24hoursflowers	English (US)	Custom	2018-09-04	In Development Analytics Edit Delete

A modal window is open over the first skill, displaying the Skill ID: **amzn1.ask.skill.b4cc820a-63bf-4622-b512-b62eb97a97be**.

Select the skill

Skills	Earnings	Payments			
Alexa Skills					Create Skill
SKILL NAME	LANGUAGE	TYPE	MODIFIED	STATUS	ACTIONS
 acme insurance <small>View Skill ID</small>	English (US)	Custom	2018-09-04	● In Development	Analytics Edit Delete
 24hoursflowers <small>View Skill ID</small>	English (US)	Custom	2018-09-04	● In Development	Analytics Edit Delete

Configure your skill

The screenshot shows the Alexa developer console interface. The top navigation bar includes links for 'Your Skills', 'acme insurance', 'Build' (which is the active tab), 'Test', 'Distribution', 'Certification', and 'Analytics'. A language selector dropdown shows 'English (US)'. Below the tabs are two buttons: 'Save Model' and 'Build Model'. On the left, a sidebar titled 'CUSTOM' contains sections for 'Interaction Model' and 'Invocation'. The 'Invocation' section is currently selected and expanded, showing a list of intents: 'Intents (5)' which includes 'Built-In Intents (5)' with items like 'AMAZON.FallbackIntent', 'AMAZON.CancelIntent', 'AMAZON.HelpIntent', and 'AMAZON.StopIntent'. To the right of the sidebar, the main content area has a heading 'Invocation' with a sub-section explaining what users say to begin an interaction. It shows a sample interaction: 'User: Alexa, ask daily horoscopes for the horoscope for Gemini'. Below this, a field labeled 'Skill Invocation Name' contains the value 'acme insurance'.

Creating intents

The screenshot shows the Oracle AI Model Builder interface for creating intents. The top navigation bar includes links for 'Your Skills', 'acme insurance', 'Build', 'Test', 'Distribution', 'Certification', 'Analytics', and 'Feedback forum'. The 'Build' tab is selected.

The left sidebar menu includes 'English (US)', 'CUSTOM', 'Interaction Model', 'Invocation', 'Intents (2)', 'Slot Types (1)', 'JSON Editor', 'Interfaces', and 'Endpoint'.

The main workspace displays the 'Intents / commandBot' section. It shows 'Sample Utterances (2)' with entries: 'What might a user say to invoke this intent?', 'anything', and 'do something'. There are 'Bulk Edit' and 'Export' buttons for these utterances.

The 'Intent Slots (1)' section lists a single slot: 'name' (Order: 1, Slot Type: AMAZON.LITERAL). There are 'Edit Dialog' and 'Delete' buttons for this slot.

Custom Alexa skill – add intent via JSON editor

The screenshot shows the AWS Alexa Skills Kit (ASK) console interface. The top navigation bar includes 'Your Skills', 'acme insurance', 'Build' (selected), 'Test', 'Distribution', 'Certification', and 'Analytics'. A language dropdown shows 'English (US)'. Below the navigation are 'Save Model' and 'Build Model' buttons.

The left sidebar menu includes 'CUSTOM', 'Interaction Model', 'Invocation', 'Intents (2)', 'Slot Types (1)', 'JSON Editor' (selected), 'Interfaces', 'Endpoint', and 'Intent History'.

The 'Intents' section shows two intents: 'commandBot' and 'Built-In Intents (1)'. The 'commandBot' intent has a red box around its name. Its details show a 'name' slot with a red box around it and a sample utterance 'AMAZON.StopIntent'.

The 'Slot Types' section shows one type: 'AMAZON.LITERAL'.

The 'JSON Editor' section contains the following JSON code:

```
1  {
2   "interactionModel": {
3     "languageModel": {
4       "invocationName": "acme insurance",
5       "intents": [
6         {
7           "name": "AMAZON.StopIntent",
8           "samples": []
9         }
10      ],
11      "name": "commandBot",
12      "slots": [
13        {
14          "name": "name",
15          "type": "AMAZON.LITERAL"
16        }
17      ],
18      "samples": [
19        "anything",
20        "do something"
21      ]
22    },
23    "types": []
24  }
```

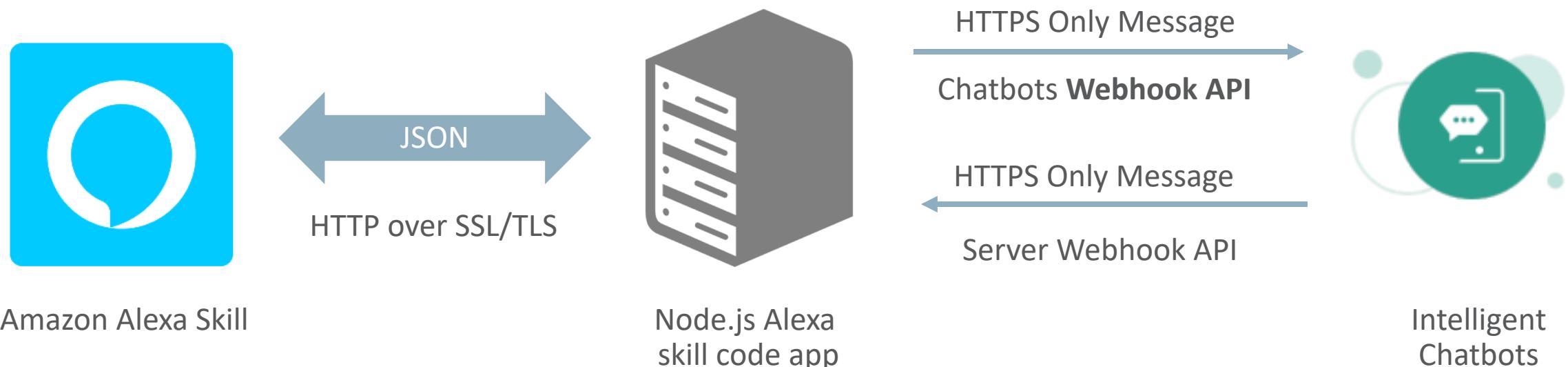
A red box highlights the entire JSON object structure.

Alexa Integration

Building the Alexa Skill Code App

About Alexa skill code app

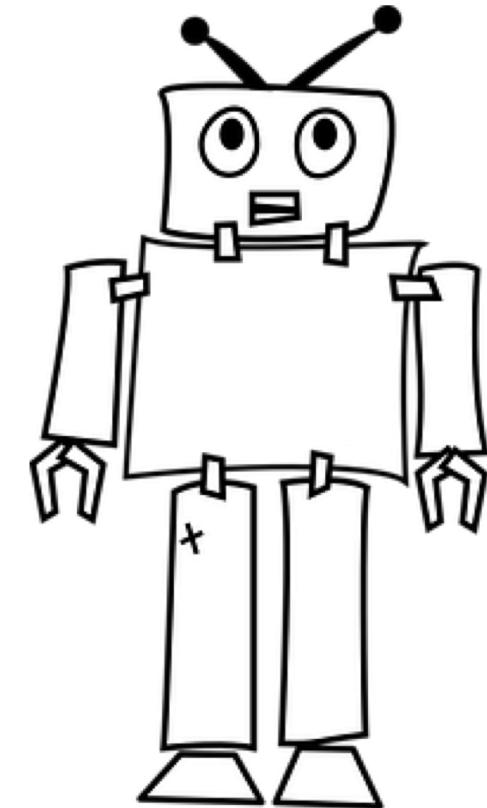
- Parses HTTP JSON requests from the Alexa platform
- Communicates with Intelligent Chatbot via **Webhook**
- Builds the JSON response that is consumed by an Alexa-compatible device



Alexa skill code app – Request types handled

- Launch Request
 - Is called when the user invokes the skill with the invocation name
 - Invokes a new session
- Intent Request
 - Is called when the user speaks a command that maps to intent
 - Sends web hook messages to the bot
 - Receives callback message from Webhook channel
 - Compose text response to Alexa
- AMAZON Specific Intents
 - Handle AMAZON.StopIntent to end the session

You need a Node.js container can be accessed from HTTPS



Note

- Keep a note of your Web URL along with the application name
 - Required to configure
 - Alexa Service Skill endpoint
 - Bot's Channel Web socket Outgoing URL



niaqnaalexabot

Version: 9.0

Last Deployed On: Aug 24, 2018 9:02:23 AM GMT

Runtime: Node 6.12.2

Created On: Jun 21, 2018 10:50:13 AM GMT

URL: <https://niaqnaalexabot-ocloud109.apaas.us2.oraclecloud.com>

Alexa skill code app - configuration

Alexa POST endpoint

```
app.locals.endpoints.push({  
    name: 'alexa',  
    method: 'POST',  
    endpoint: '/alexa/app'  
});
```

Service Endpoint Type

Select how you will host your skill's service endpoint.

- AWS Lambda ARN ?
(Recommended)
- HTTPS ?

Default Region ?
(Required)

hiaqnaalexabot-ocloud109.apaas.us2.oraclecloud.com/apps/alexa-singleBot/alexa/app

Alexa skill code app- configuration

Bot Webhook POST endpoint

```
app.locals.endpoints.push({  
    name: 'Webhook',  
    method: 'POST',  
    endpoint:  
    '/singleBotWebhook/messages'  
});
```

Create Channel X

* Name

Description

Channel Type

Platform Version

* Outgoing Webhook URI (highlighted)

Session Expiration (minutes) Default

Channel Enabled

Create

Alexa skill code app- configuration

- Download *alexa-singleBot* sample, update app.js for the following:
 - Amazon Skill ID
 - Channel Secret Key
 - Channel URL

```
//replace these settings to point to your webhook channel
var metadata = {
  allowConfigUpdate: true,
  waitForMoreResponsesMs: 200,
  amzn_appId: "amzn1.ask.skill.b6a6XXXXX3f09-922ddc2aded9",
  channelSecretKey: '8SJBdyXXXXXXjsyq53uiPIV5fPVewu',
  channelUrl: 'http://botconn:8000/connectors/v1/tenants/chatbot-tenant/listeners/webhook/channels/8C9689BB-2105'
};
```

Web Server app - configuration

- Download *alexa-singleBot* sample, update app.js for the following:
 - Amazon Skill ID
 - Channel Secret Key
 - Channel URL

```
//replace these settings to point to your webhook channel
var metadata = {
  allowConfigUpdate: true,
  waitForMoreResponsesMs: 200,
  amzn_appId: "amzn1.ask.skill.b6a6XXXXX3f09-922ddc2aded9",
  channelSecretKey: '8SJBDyXXXXXXXXjsyq53uiPIV5fPVewu',
  channelUrl: 'http://botconn:8000/connectors/v1/tenants/chatbot-tenant/listeners/webhook/channels/8C9689BB-2105'
};
```

Alexa skill code app - configuration

- Download *alexa-singleBot* sample, update app.js for the following:

- **Amazon Skill ID** Obtained after successful creation of Amazon Skill
- **Channel Secret Key**
- **Channel URL** Obtained after successful creation Webhook channel in Oracle Digital Assistant

```
//replace these settings to point to your webhook channel
var metadata = {
  allowConfigUpdate: true,
  waitForMoreResponsesMs: 200,
  amzn_appId: "amzn1.ask.skill.b6a6XXXXX3f09-922ddc2aded9",
  channelSecretKey: '8SJBDyXXXXXXXXjsyq53uipIV5fPVewu',
  channelUrl: 'http://botconn:8000/connectors/v1/tenants/chatbot-tenant/listeners/webhook/channels/8C9689BB-2105'
};
```

Configure skill app endpoint

The screenshot shows the Alexa Skills Kit (ASK) configuration interface for a custom skill named "CUSTOM". The left sidebar lists various skill components: Interaction Model, Invocation, Intents (2), CommandBot (with command intent), Built-In Intents (1) (AMAZON.StopIntent), Slot Types (1) (LITERAL), JSON Editor, Interfaces, and Endpoint (which is selected and highlighted in blue). The main content area is titled "Endpoint" and contains a descriptive message about receiving POST requests and links to Lambda endpoints and HTTPS requirements. Below this is a section titled "Service Endpoint Type" with two options: "AWS Lambda ARN" (Recommended) and "HTTPS" (selected). A "Default Region" field is set to "https://niaqnaalexabot-ocloud109.apaas.us2.oraclecloud.com/apps/alexa-singleBot/ale". A note below states: "My development endpoint is a sub-domain of a domain that has a wildcard certifica...".

CUSTOM

Interaction Model

Invocation

Intents (2)

+ Add

CommandBot

command

Built-In Intents (1)

AMAZON.StopIntent

Slot Types (1)

+ Add

LITERAL

JSON Editor

Interfaces

Endpoint

Endpoint

The Endpoint will receive POST requests when a user interacts with your Alexa Skill. The request body contains parameters that your service can use to perform logic and generate a JSON-formatted response. Learn more about Lambda endpoints [here](#). You can host your own HTTPS web service endpoint as long as the service meets the requirements described [here](#).

Service Endpoint Type

Select how you will host your skill's service endpoint.

AWS Lambda ARN (Recommended)

HTTPS (Required)

Default Region (Required)

`https://niaqnaalexabot-ocloud109.apaas.us2.oraclecloud.com/apps/alexa-singleBot/ale`

My development endpoint is a sub-domain of a domain that has a wildcard certifica...

Configure skill app endpoint

The screenshot shows the 'Endpoint' configuration page for a custom Alexa skill. The left sidebar lists various skill components: Interaction Model, Invocation, Intents (2), CommandBot (with command intent), Built-In Intents (1) (AMAZON.StopIntent), Slot Types (1), LITERAL, JSON Editor, Interfaces, and Endpoint (which is selected and highlighted in blue). The main area is titled 'Endpoint' and contains a descriptive text box explaining the purpose of the endpoint. Below this, the 'Service Endpoint Type' is set to 'Service Endpoint'. A 'Service Endpoint' input field contains the URL <https://niaqnaalexabot-ocloud109.apaas.us2.oraclecloud.com/apps/alexa-singleBot/alexa/app>. A 'Default Region' dropdown is set to '(Required)' with the value <https://niaqnaalexabot-ocloud109.apaas.us2.oraclecloud.com/apps/alexa-singleBot/alexa/app>. A note below the region dropdown states: 'My development endpoint is a sub-domain of a domain that has a wildcard certificate...'.

Alexa Integration

Configuration in Oracle Intelligent Bots

Oracle Digital Assistant - create channel

Create Channel X

*** Name** AlexaWebhook

Description Alexa integration

Channel Type Webhook

Platform Version 1.1 (Conversation Model)

*** Outgoing Webhook URI** <https://niaqnaalexabot-ocloud109.apaas.us2.oraclecloud.com/apps/a>

Session Expiration (minutes) 60 ▼ ▲ Default

Channel Enabled

Create

Oracle Digital Assistant - create channel

Create Channel X

* Name

Description

Channel Type ▼

Platform Version ▼

* Outgoing Webhook URI

Session Expiration (minutes) ▼ ▲ Default

Channel Enabled

Create

Outgoing Webhook URI:
<https://niaqnaalexabot-ocloud109.apaas.us2.oraclecloud.com/apps/alexa-singleBot/singleBotWebhook/messages>

Oracle Digital Assistant - create channel

+ Channel

Reset Sessions

Alexa	X
FacebookChannel	X
System_Bot_Test	X
WebChannel	X

* Name: Alexa

Description: *Optional short description for this channel*

Channel Type: Webhook

Platform Version: 1.1 (Conversation Model)

Outgoing Webhook URI: <https://niaqnaalexabot-ocloud109.apaas.us2.oraclecloud.com/apps/alexa-singleBot/singleBotWe>

Secret Key: XOkxH9NJNeOzChq6AjNWBa8AyHTOBx4a [Reset](#)

Webhook URL: <https://PMTEAMbmxp-pmamcenas3.mobile.ocp.oraclecloud.com:443/connectors/v1/tenants/idcs-8aa35a06ae724eba815f48b66c3835c8/listeners/webhook/channels/76766271-0BE0-43C3-ACCB-2992439A0596>

Session Expiration (minutes): 60 ▾ ▾ Default

Channel Enabled:

Oracle Digital Assistant - create channel

+ Channel

Reset Sessions

Channel	X
Alexa	X
FacebookChannel	X
System_Bot_Test	X
WebChannel	X

Secret Key and Webhook URL will be passed in skill code configuration

* Name: Alexa

Description: Optional short description for this channel

Channel Type: Webhook

Platform Version: 1.1 (Conversation Model)

Outgoing Webhook URI: https://niaqnaalexabot-ocloud109.apaas.us2.oraclecloud.com/apps/alexa-singleBot/singleBotWe

Secret Key: XOlkxH9NJNeOzChq6AjNWBa8AyHTOBx4a [Reset](#)

Webhook URL: https://PMTEAMbmxp-pmamcenas3.mobile.ocp.oraclecloud.com:443/connectors/v1/tenants/idcs-8aa35a06ae724eba815f48b66c3835c8/listeners/webhook/channels/76766271-0BE0-43C3-ACCB-2992439A0596

Session Expiration (minutes): 60 ▾ ▾ Default

Channel Enabled:



Oracle Digital Assistant Hands-On

TBD

Integrated Cloud Applications & Platform Services

ORACLE®