

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

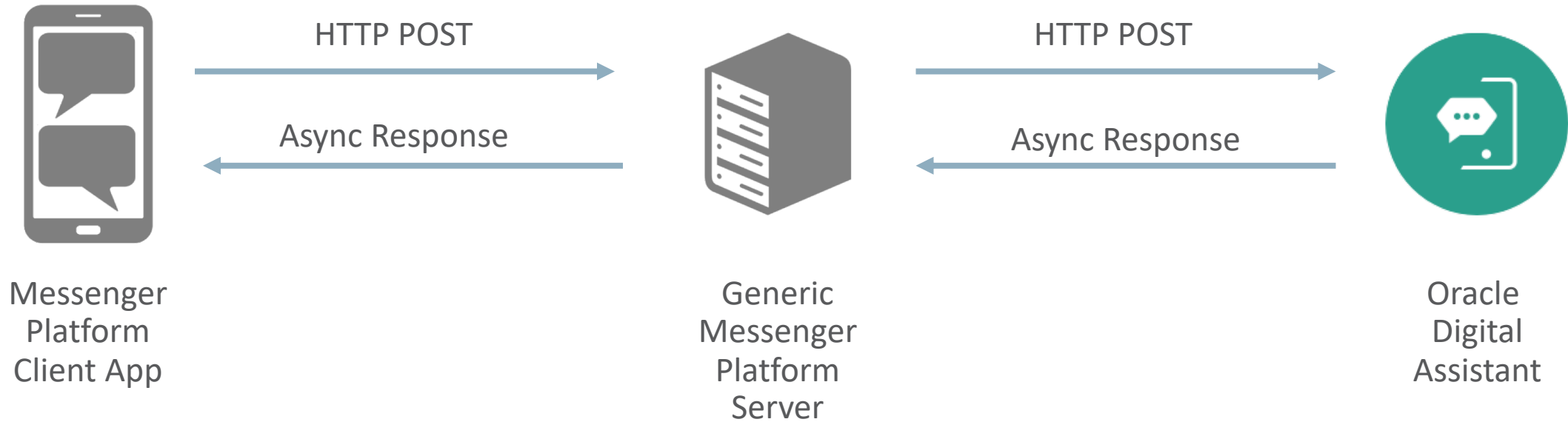
- 1 Overview
- 2 Creating webhook clients with Node.js
- 3 Voice integration with Alexa

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/>

300 commits 3 branches 32 releases 2 contributors View license

Branch: master New pull request Find file Clone or download

Clone with HTTPS
Use Git or checkout with SVN using the web URL.
<https://github.com/oracle/bots-node-s>

Open in Desktop **Download ZIP**

bin	Bugfix #3 - Spawn ENOENT on windows
common	Removed joi-browser for browser scenario.
config	Addresses initial SCS analysis
examples	restored starter readme too

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

sample

- index.js
- package.json
- README.md
- sample.req.json
- service.js

Create webhook client

The image shows a terminal window and a file explorer side-by-side. The terminal window, titled 'sample — -bash — 80x24', shows the following commands and 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$
```

The file explorer shows the directory structure of the 'sample' folder. The 'node_modules' folder is highlighted with a red box. The files listed are:

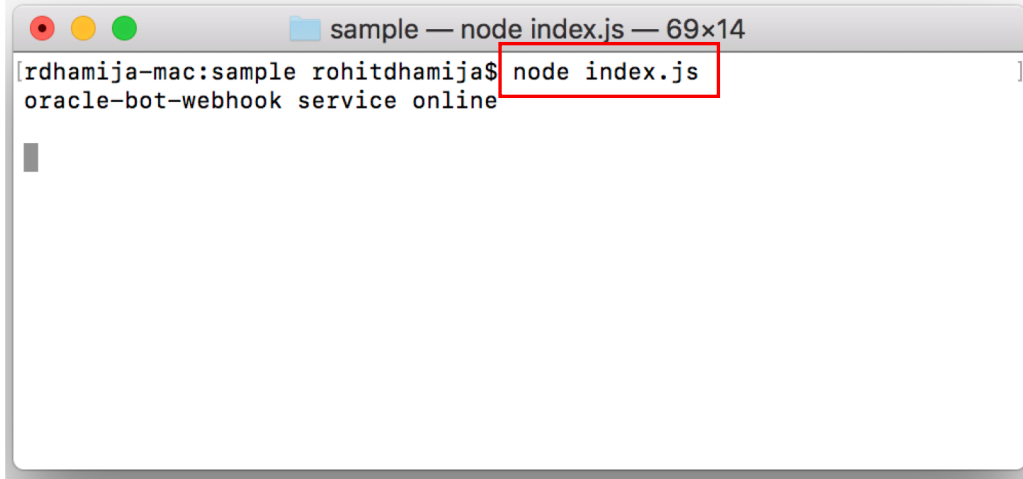
- package-lock.json
- package.json
- 2018
 - index.js
 - README.md
 - sample.req.json
 - service.js

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" is shown. The prompt is "[rdhamija-mac:sample rohitdhamija\$". The command "node index.js" has been entered and is highlighted with a red box. The output of the command is "oracle-bot-webhook service online".



localhost:3000

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

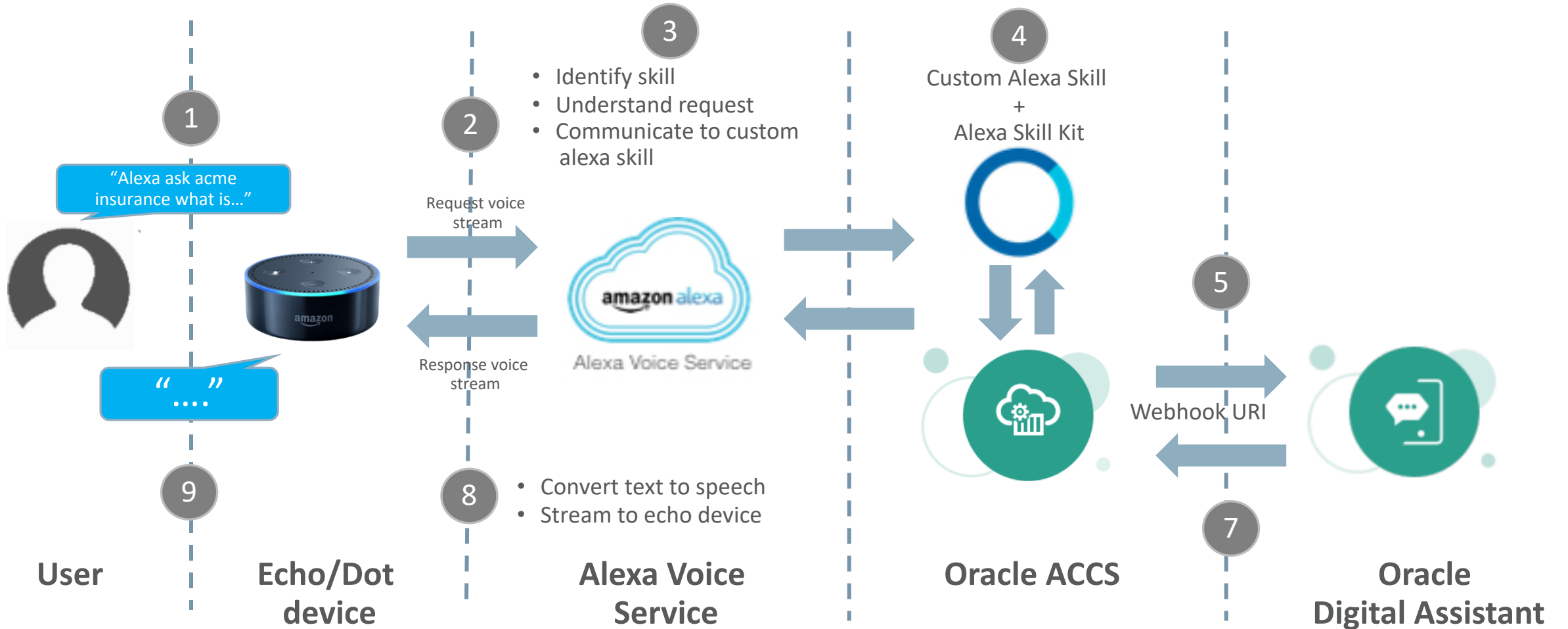


Oracle Digital
Assistant

Create
Webhook
channel



Alexa integration reference architecture



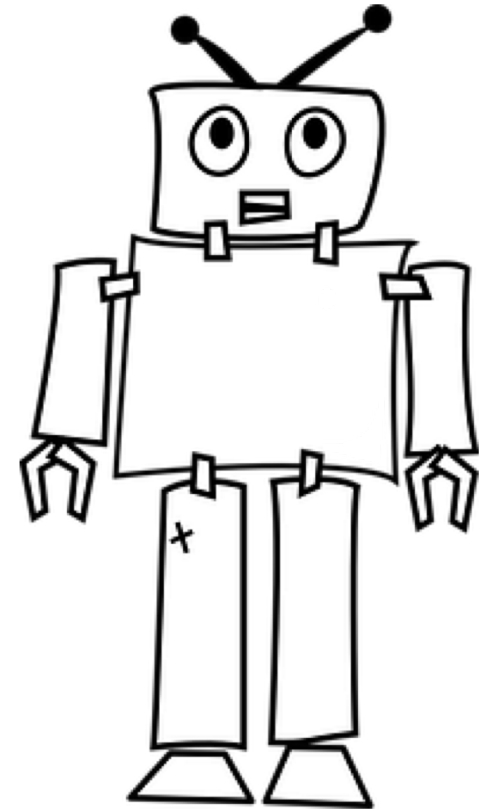
Alexa integration

Amazon Alexa

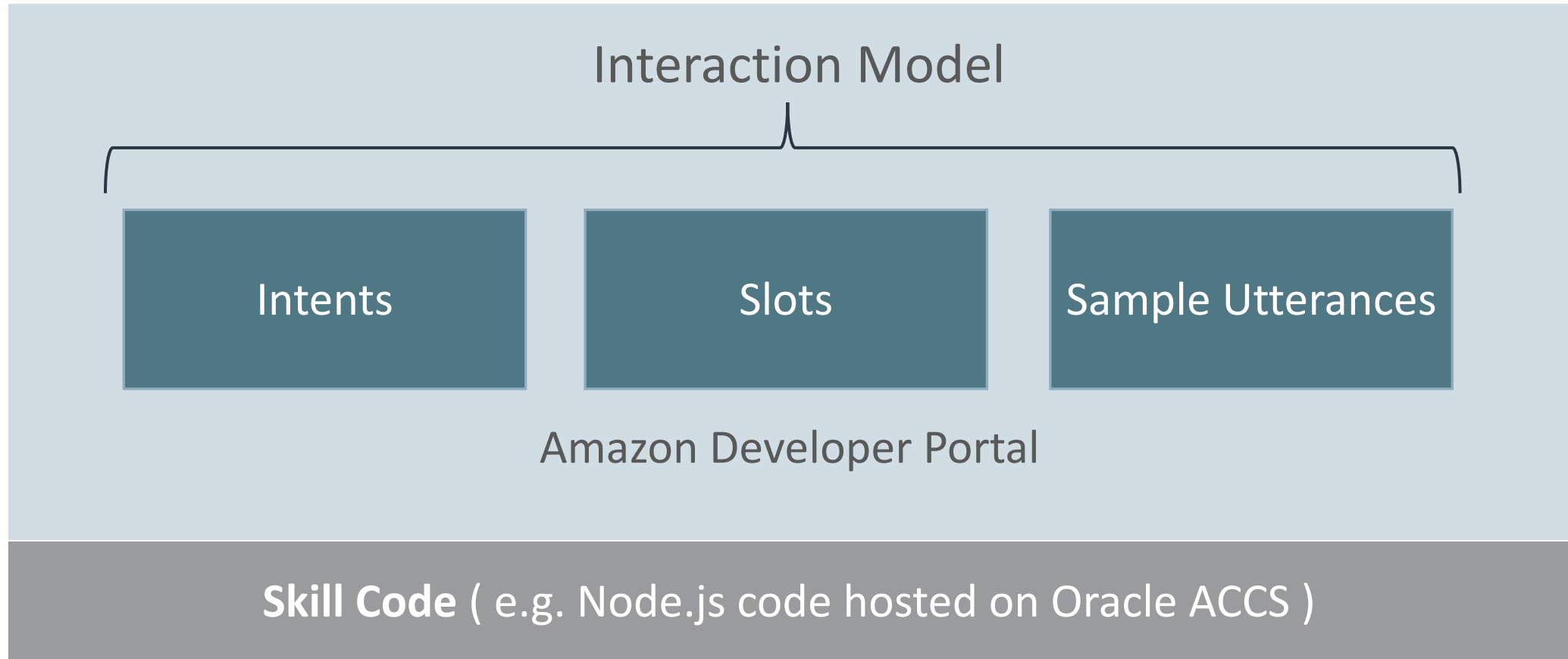
Key terminologies

- 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

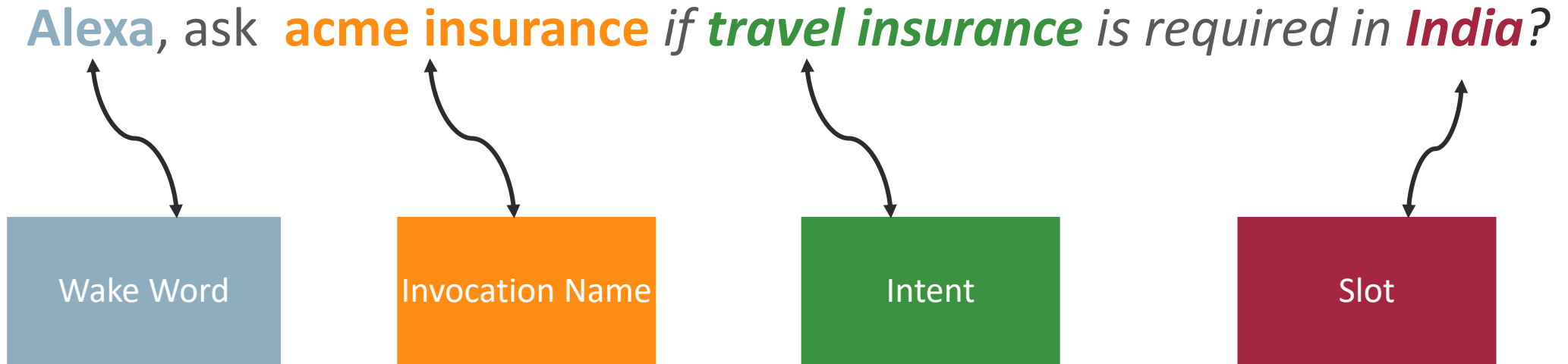
You need to develop an **Alexa Skill** to integrate Alexa as a channel with Oracle Intelligent Bots



Alexa custom skill components



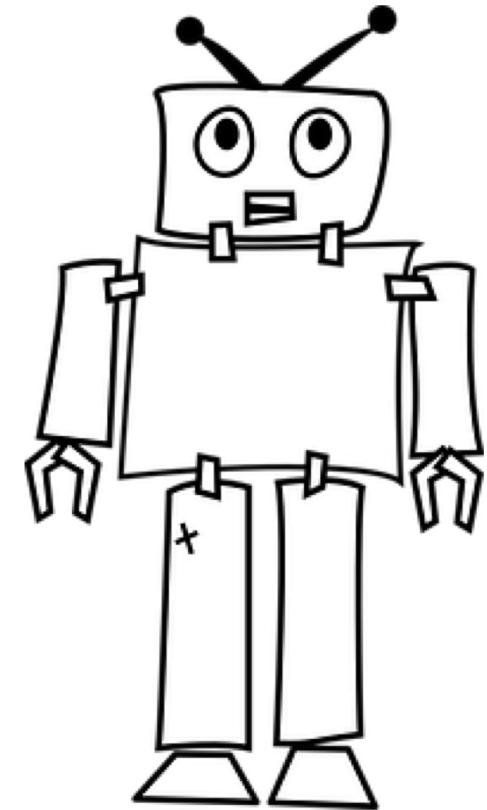
Parsing invocation name, intent and slots



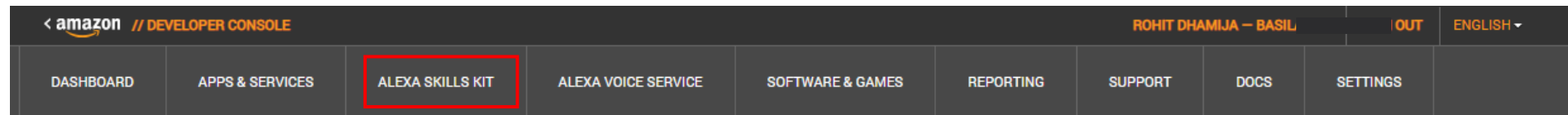
Alexa Integration

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 View Skill ID	English (US)	Custom	2018-09-04	● In Development	Analytics Edit Delete

Creating a skill

Create a new skill

Cancel

Create 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

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

SELECTED

Flash Briefing

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

"Alexa, what's in the news?"

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.

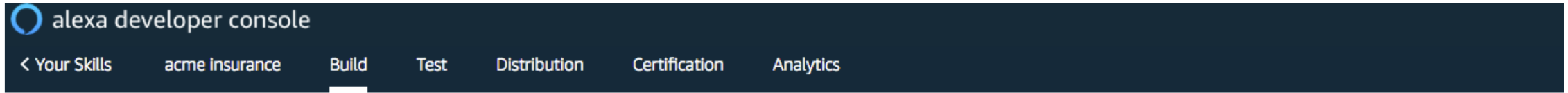
"Alexa, turn on the kitchen lights"

Video

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

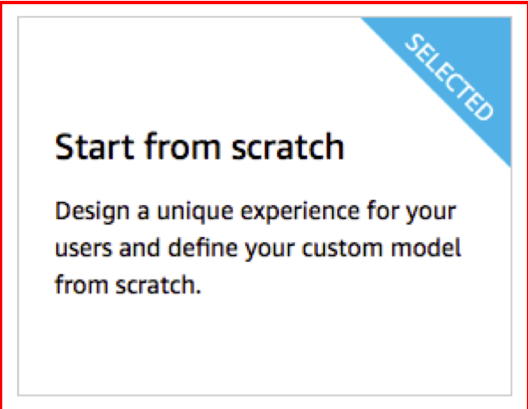
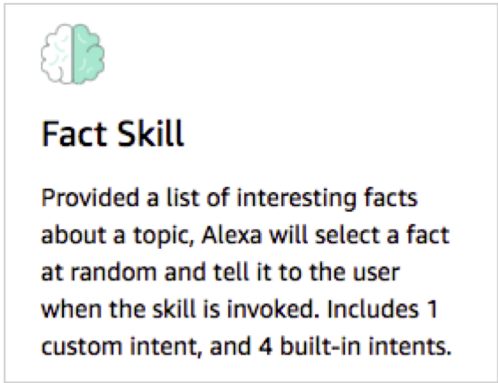
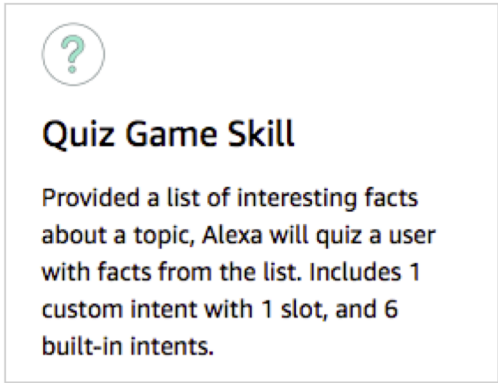
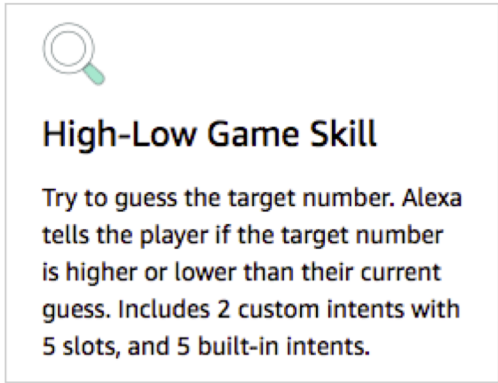
"Alexa, play Interstellar"

Creating a skill – choose a template



Choose a template

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

 <p>Start from scratch</p> <p>Design a unique experience for your users and define your custom model from scratch.</p> <p><i>SELECTED</i></p>	 <p>Fact Skill</p> <p>Provided 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.</p>	 <p>Quiz Game Skill</p> <p>Provided 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.</p>	 <p>High-Low Game Skill</p> <p>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.</p>
--	---	--	--

Get the skill ID



Skills

Earnings

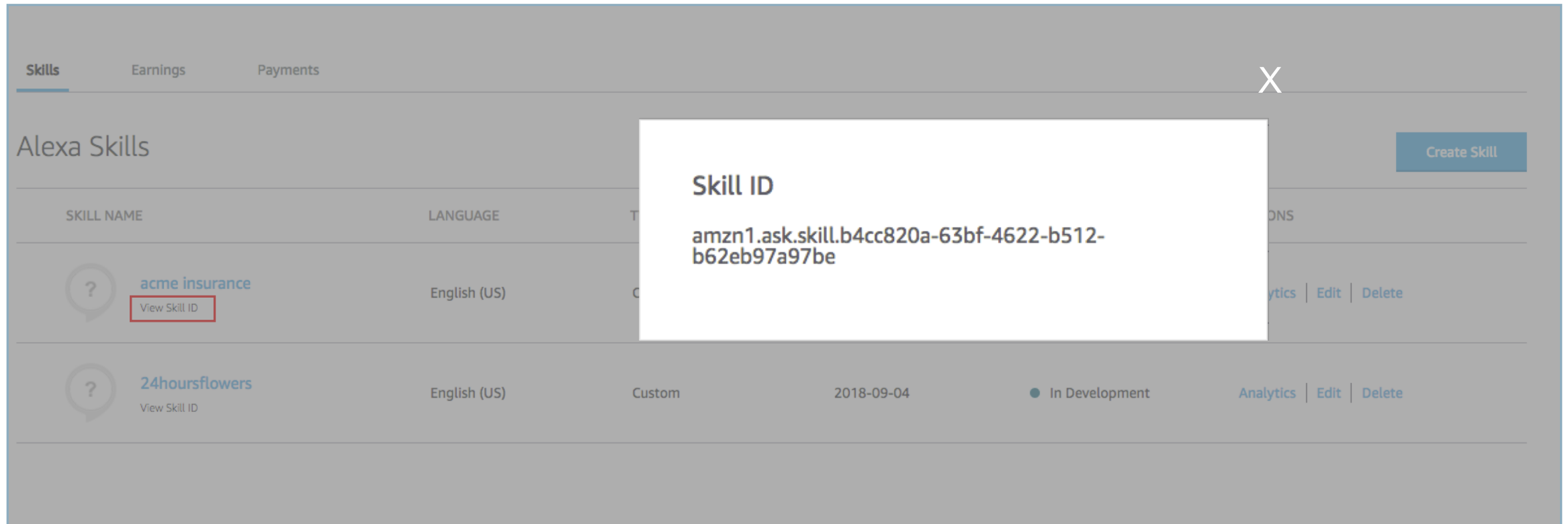
Payments

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 console interface. At the top, there are tabs for 'Skills', 'Earnings', and 'Payments'. A modal window is open in the center, displaying the 'Skill ID' for the selected skill. The skill ID is: `amzn1.ask.skill.b4cc820a-63bf-4622-b512-b62eb97a97be`. The background shows a table of skills with columns for Skill Name, Language, and other details. The 'acme insurance' skill is highlighted, and its 'View Skill ID' button is circled in red.

SKILL NAME	LANGUAGE	TYPE	CREATED	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

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. At the top, the navigation bar includes 'Your Skills', 'acme insurance', 'Build' (selected), 'Test', 'Distribution', 'Certification', and 'Analytics'. Below the navigation bar, there is a language selector set to 'English (US)' and two buttons: 'Save Model' and 'Build Model'. The left sidebar is titled 'CUSTOM' and contains a tree view with 'Interaction Model' and 'Invocation' (selected). Under 'Invocation', there is a section for 'Intents (5)' with an 'Add' button. Below this, a list of built-in intents is shown: 'AMAZON.FallbackIntent', 'AMAZON.CancelIntent', 'AMAZON.HelpIntent', and 'AMAZON.StopIntent'. The main content area is titled 'Invocation' and contains the following text: 'Users say a skill's invocation name to begin an interaction with a particular custom skill. For example, if the invocation name is "daily horoscopes", users can say:'. Below this text is a green-bordered box containing the example user utterance: 'User: Alexa, ask daily horoscopes for the horoscope for Gemini'. At the bottom of the main content area, there is a field labeled 'Skill Invocation Name' with a help icon, containing the text 'acme insurance'.

Creating intents

The screenshot shows the Amazon Lex console interface for building a skill. The top navigation bar includes 'Your Skills', 'acme Insurance', 'Build', 'Test', 'Distribution', 'Certification', 'Analytics', and 'Feedback forum'. The left sidebar contains navigation options: 'CUSTOM', 'Interaction Model', 'Invocation', 'Intents (2)' (with an 'Add' button), 'commandBot' (selected), 'name', 'Built-In Intents (1)' (including 'AMAZON.StopIntent'), 'Slot Types (1)' (with an 'Add' button, including 'AMAZON.LITERAL'), 'JSON Editor', 'Interfaces', and 'Endpoint'. The main content area is titled 'Intents / commandBot' and includes 'Save Model' and 'Build Model' buttons. Below the title, there are 'Sample Utterances (2)' with a 'Bulk Edit' and 'Export' link. A text input field contains 'What might a user say to invoke this Intent?'. Below this, two sample utterances are listed: 'anything' and 'do something', each with a delete icon. A pagination indicator shows '1 - 2 of 2'. The 'Intent Slots (1)' section contains a table with the following data:

ORDER	NAME	SLOT TYPE	ACTIONS
1	name	AMAZON.LITERAL	Edit Dialog Delete

Custom Alexa skill – add intent via JSON editor

The screenshot shows the Amazon Alexa Developer Console interface for editing a custom skill. The left sidebar contains a navigation menu with the following items: **Interaction Model**, **Invocation**, **Intents (2)** (with a red box around 'commandBot'), **Built-In Intents (1)** (with 'AMAZON.StopIntent'), **Slot Types (1)** (with 'AMAZON.LITERAL'), **JSON Editor** (highlighted in blue), **Interfaces**, **Endpoint**, and **Intent History**. The main content area is titled 'JSON Editor' and includes a link to learn more about the schema definition. Below the link is a dashed box with the text 'Drag and drop a .json file'. The JSON editor displays the following code:

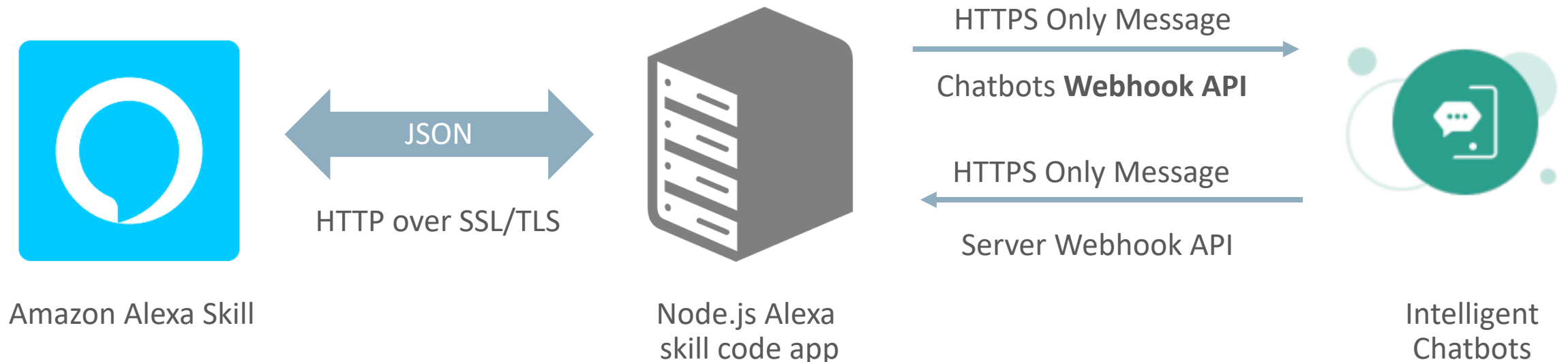
```
1 {
2   "interactionModel": {
3     "languageModel": {
4       "invocationName": "acme insurance",
5       "intents": [
6         {
7           "name": "AMAZON.StopIntent",
8           "samples": []
9         },
10        {
11          "name": "commandBot",
12          "slots": [
13            {
14              "name": "name",
15              "type": "AMAZON.LITERAL"
16            }
17          ],
18          "samples": [
19            "anything",
20            "do something"
21          ]
22        }
23      ]
24    },
25    "types": []
26  }
27 }
```

Alexa Integration

Building the Alexa Skill Code App

About Alexa skill code app

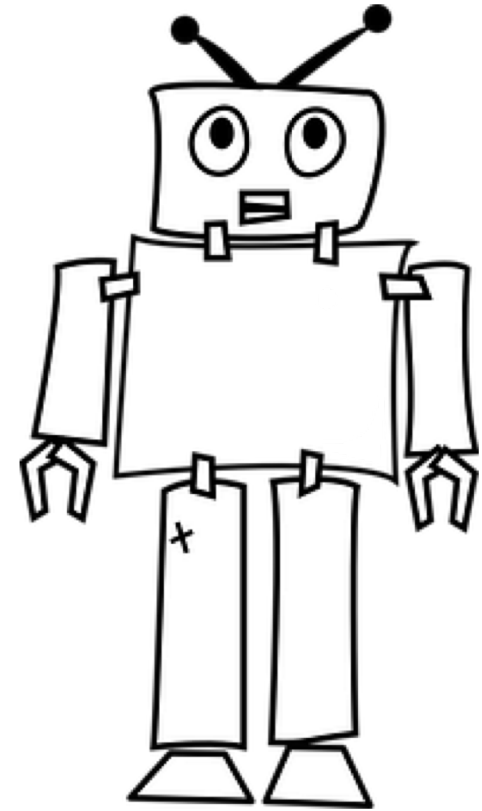
- Parses HTTP JSON requests from the Alexa platform
- Communicates with Intelligent Chatbot via **Webhook**
- Builds the JSON response that is consumed by an Alexa-compatible device



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

* Name: AlexaWebhook

Description: Optional short description for this channel

? Channel Type: Webhook

? Platform Version: 1.1 (Conversation Model)

? * Outgoing Webhook URI: .oraclecloud.com/apps/alexa-singleBot/singleBotWebhook/messages

Session Expiration (minutes): 60 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: '8SJBdyXXXXXXjsyq53uipIV5fPVewu',
  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

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 ?

Default Region ?
(Required)

Configure skill app endpoint

CUSTOM

- Interaction Model
- Invocations
- 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

Service Endpoint: (Recommended)

HTTPS 🕒

Default Region 🕒 (Required)

🗑

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

Alexa Integration

Configuration in Oracle Intelligent Bots

Oracle Digital Assistant - create channel

Create Channel ✕

*** Name**

Description

? Channel Type

? Platform Version

? * Outgoing Webhook URI

Session Expiration (minutes) Default

Channel Enabled

Oracle Digital Assistant - create channel

Create Channel ✕

*** Name**

Description

Channel Type

Platform Version

*** Outgoing Webhook URI**

Session Expiration (minutes) Default

Channel Enabled





Outgoing Webhook URI:

<https://niaqnaalexabot-ocloud109.apaas.us2.oraclecloud.com/apps/alexasingleBot/singleBotWebhook/messages>

Oracle Digital Assistant - create channel

+ Channel

Reset Sessions

-  Alexa ×
-  FacebookChannel ×
-  System_Bot_Test ×
-  WebChannel ×

* Name

Description

Channel Type

? Platform Version

? * Outgoing Webhook URI

Secret Key [Reset](#)

Webhook URL

Session Expiration (minutes) Default

Channel Enabled

Oracle Digital Assistant - create channel

+ Channel

Reset Sessions

- Alexa
- FacebookChannel
- System_Bot_Test
- WebChannel

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/alexasingleBot/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 [v](#) [^](#) Default

Channel Enabled:



Oracle Digital Assistant Hands-On

TBD

Integrated Cloud

Applications & Platform Services

ORACLE®