

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 ➤ Creating a webhook channel

Topic agenda

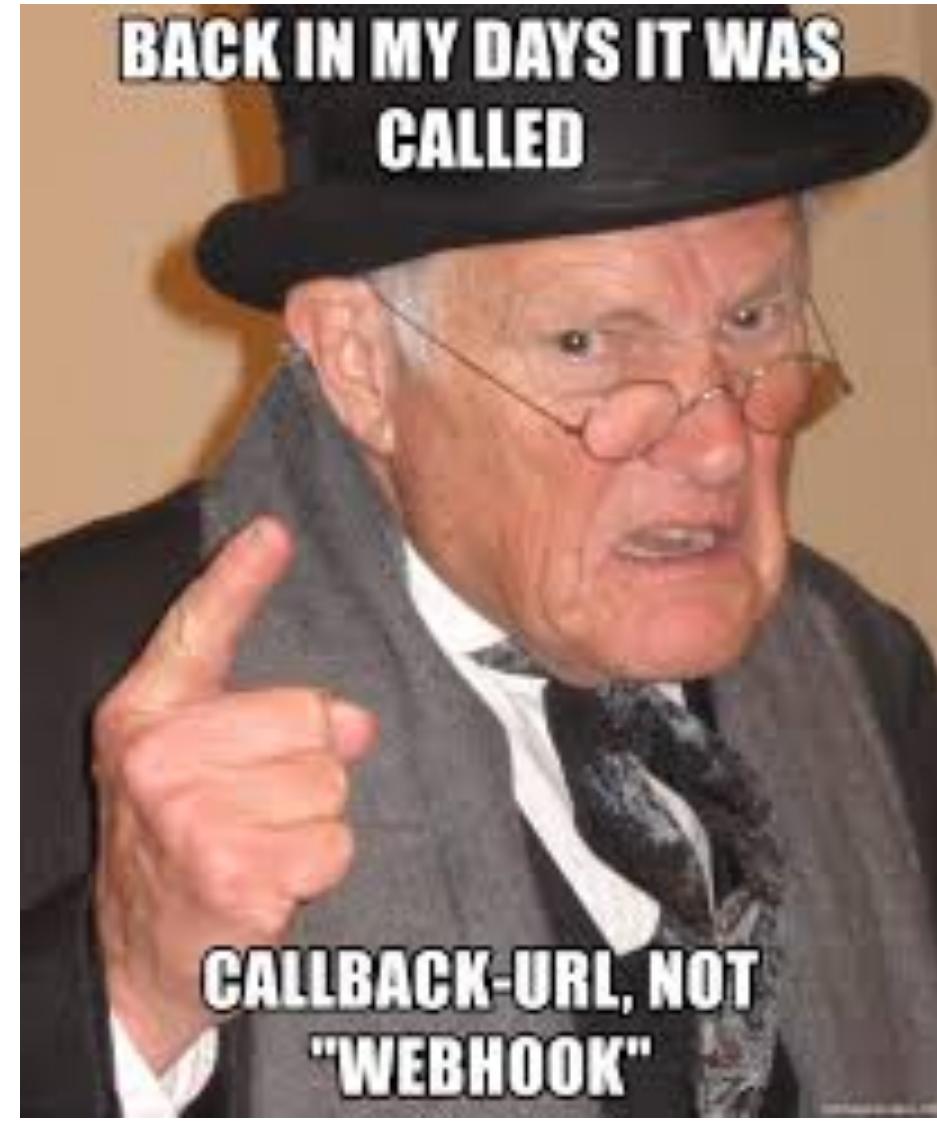
1 ➤ Overview

2 ➤ Creating webhook clients with Node.js

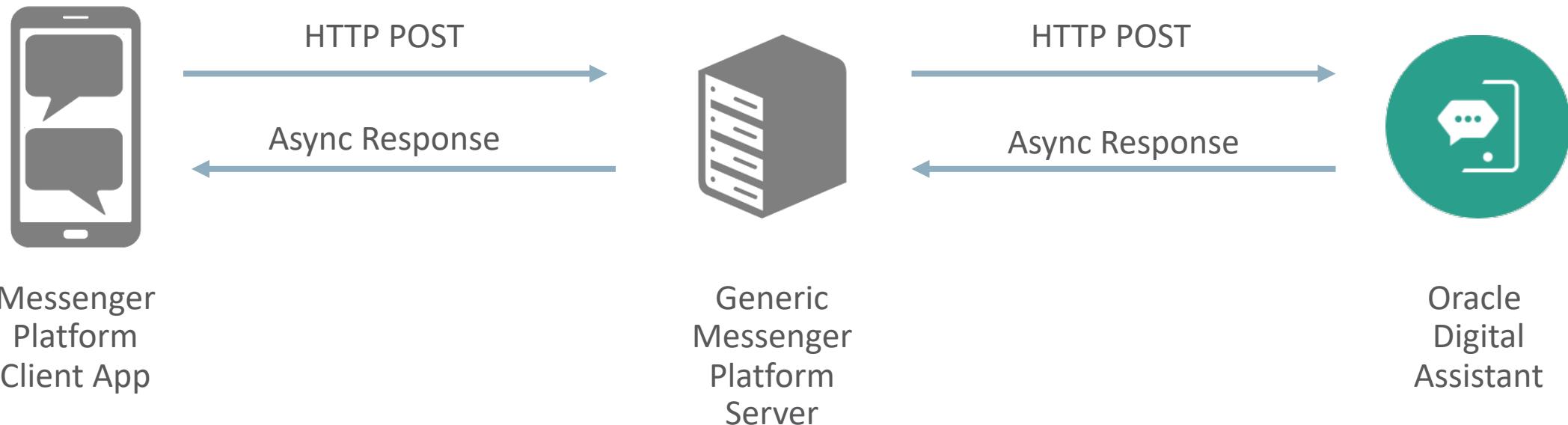
3 ➤ Creating a webhook channel

About webhooks and their use with chatbots

- Programming interfaces implemented by a system on the web
 - Allows other programs to subscribe and receive automated notification
- Accessing chatbots using Webhooks
 - Users access chatbots through a messenger
 - Webhook are "client adapters" that dispatch and manage conversation between messenger channels and chatbots
- Webhooks and Oracle Digital Assistant
 - You use webhooks to access Oracle Digital Assistant for messenger channels that there is no native channel support provided for



Generic HTTP webhook channel



Generic HTTP webhook support

- 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 Oracle Digital Assistant generates and uses a secret key
 - 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

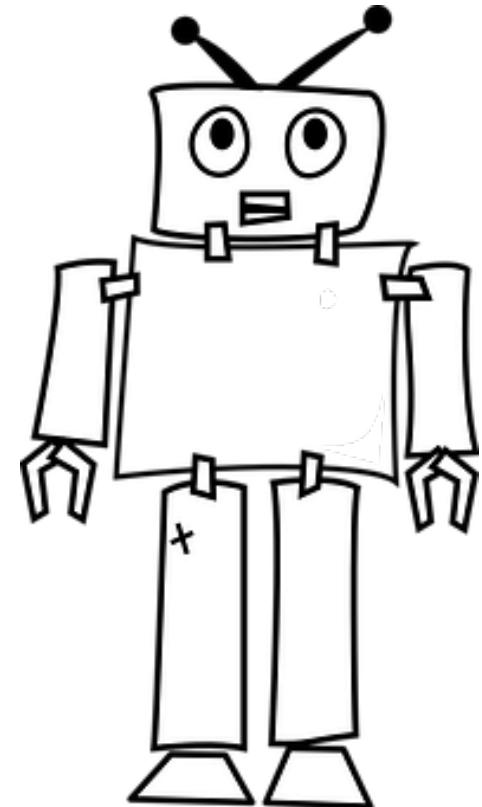
3

Creating a webhook channel

Building steps for Oracle Digital Assistant

- Download Oracle Bots Node.js SDK
- Create a webhook client project
- Implement webhook client code for messages received from messenger
- Create a webhook channel in Oracle Digital Assistant
 - Associate channel with digital assistant or skill
- Reference webhook channel in webhook client

The **Oracle Bots Node.js SDK** makes
it easy to build webhook clients for
Oracle Digital Assistant.



Oracle Bots Node.js SDK

<https://github.com/oracle/bots-node-sdk>

- Bots Node.js SDK functionality
 - Custom component development
 - Webhook client development
- Webhook development support
 - Assists in building webhook clients that dispatch between messengers and bots
 - E.g. Alexa integration
 - Generates the SHA256 signature and sets X-Hub-Signature header

Getting started with your webhook development 1 of 4

Download Ortacle Bots Node.js SDK

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

The screenshot shows the GitHub repository page for the Oracle Bots Node.js SDK. At the top, there are statistics: 300 commits, 3 branches, 32 releases, 2 contributors, and a 'View license' link. Below this, there are buttons for 'Branch: master', 'New pull request', 'Find file', and a prominent green 'Clone or download' button. To the right of the clone button is a 'Clone with HTTPS' section with a URL and a copy icon. A red box highlights the 'Download ZIP' button at the bottom right of the page.

300 commits 3 branches 32 releases 2 contributors View license

Branch: master New pull request Find file Clone or download

mvandervliet Merge branch 'develop' into 'master' ...

bin Bugfix #3 - Spawn ENOENT on windows

common Removed joi-browser for browser scenario.

config Addresses initial SCS analysis

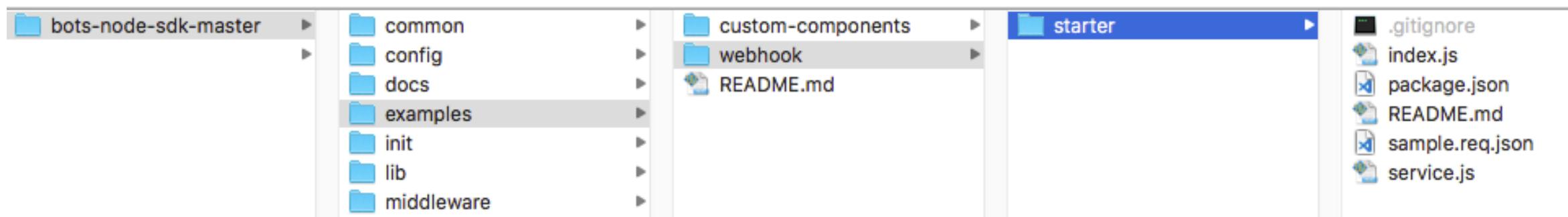
examples restored starter readme too

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**

Getting started with your webhook development 2 of 4

Extract ZIP and navigate to `bots-node-sdk-master/examples/webhook/starter` folder

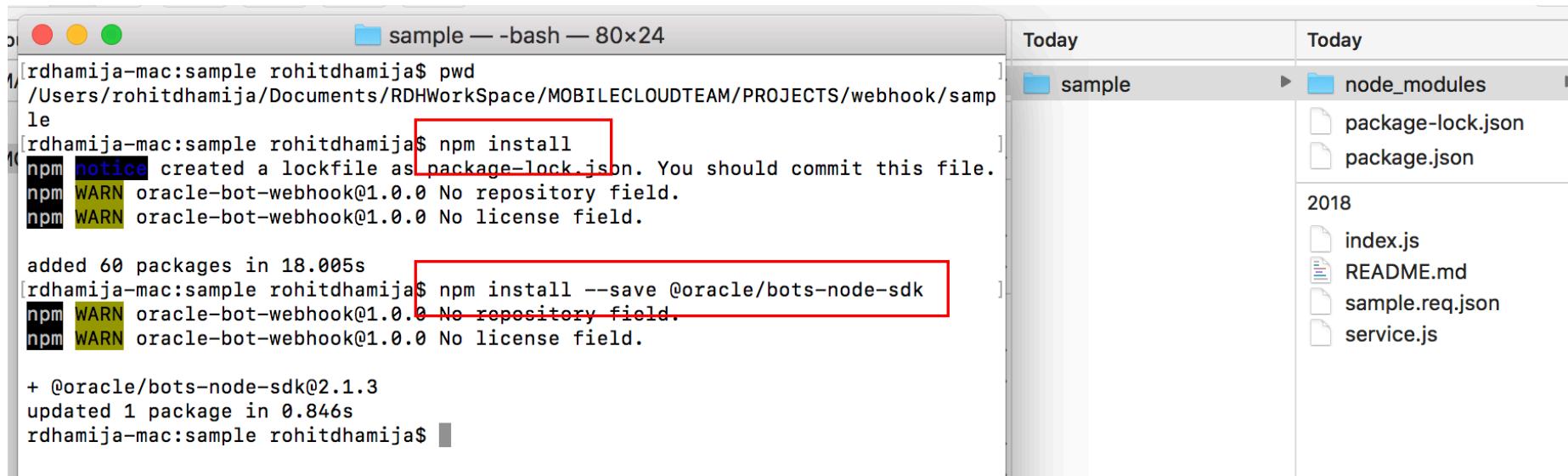


Getting started with your webhook development 3 of 4

Copy content of started folder to your custom project folder



Getting started with your webhook development 4 of 4



The image shows a Mac OS X terminal window titled "sample — bash — 80x24" and a Finder window side-by-side.

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 Explorer:

The Finder window shows the directory structure of the "sample" project:

- sample (Folder)
 - node_modules (Folder)
 - package-lock.json
 - package.json
- 2018
 - index.js
 - README.md
 - sample.req.json
 - service.js

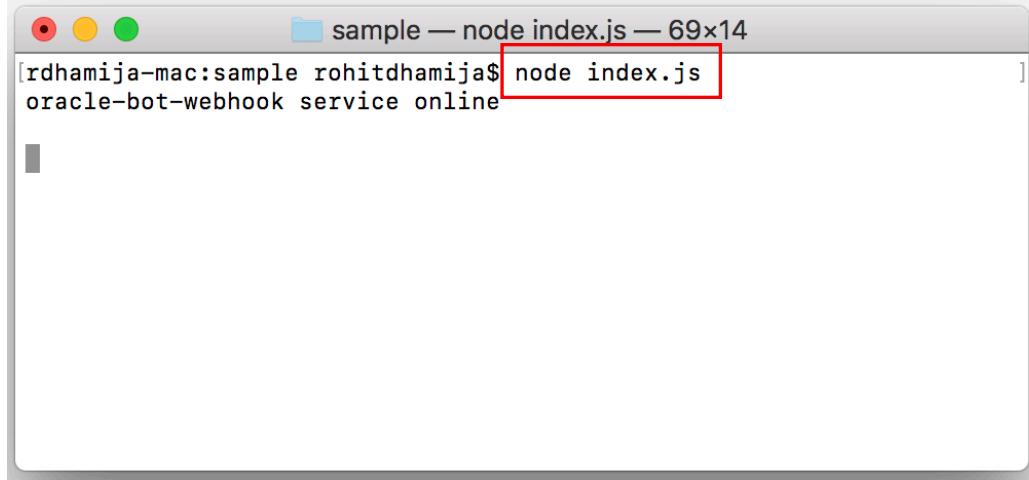
- Install project
 - npm install
 - Npm install –save @oracle/bots-node-sdk

Exploring the 'service.js' webhook client

```
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 }
```

Running the project

```
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 screenshot of a terminal window titled "sample — node index.js — 69x14". The window shows the command "node index.js" being run, followed by the output "oracle-bot-webhook service online". The command "node index.js" is highlighted with a red box.



Topic Agenda

- 1 ➔ Overview
- 2 ➔ Creating Webhook clients with Node.js (build, configure)
- 3 ➔ Creating a webhook channel

Oracle Digital Assistant – Creating a webhook channel

Create Channel X

* Name

Description

Channel Type

Platform Version

Outgoing Webhook URI

Session Expiration (minutes) Default

Channel Enabled

Create

Oracle Digital Assistant – Creating a webhook 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 – Creating a webhook 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 – Creating a webhook 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>

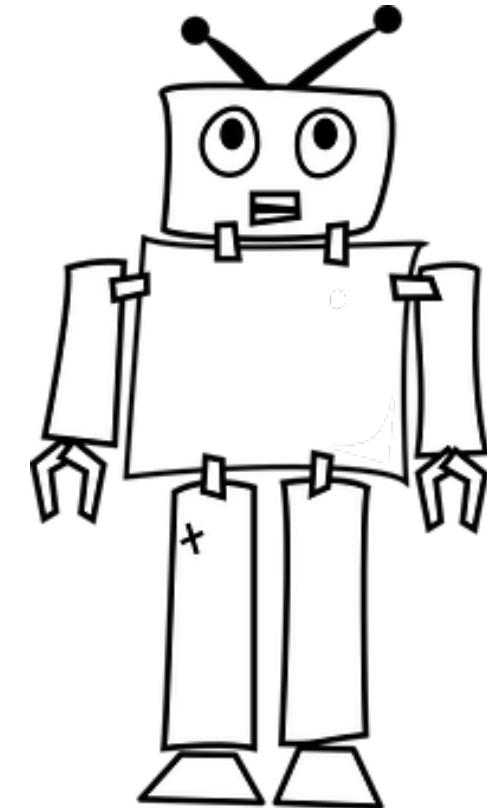
Session Expiration (minutes): 60 ▾ ▾ Default

Channel Enabled:

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

You find a complete **webhook example for Alexa** voice integration on the **Oracle TechExchange** blog

<https://blogs.oracle.com/mobile/tech-exchange>



Integrated Cloud Applications & Platform Services

ORACLE®



Oracle Digital Assistant Hands-On

TBD