

はじめてのAlexa開発キット

～RaspberryPiを使い、スキル開発から、自社製品へのAlexa搭載開発まで、PoCツール可～

実践編 抜粋版



スペクトラム・テクノロジー株式会社

<https://spectrum-tech.co.jp>

sales@spectrum-tech.co.jp

Alexa開発キット 目次

Pi運用マニュアル

1. RaspberryPiについて
2. Linux基本コマンド
3. 基本操作
4. 日常運用(ウイルススキャン、更新)

Alexa開発

- ① 全体構成
- ② メニュー
- ③ AWS設定
- ④ Amazon. co. jp設定
- ⑤ Amazon developer設定
- ⑥ Alexaスキル開発(コンソール編)
 - A) Hello world
 - B) Drink(スロット対応)
 - C) APL(画面对応)
 - D) Device address
 - E) High Low game
- ⑦ Alexaスキル開発(CLI編)
 - 概要、特徴
 - 認証設定の準備
 - 認証設定
 - 基本操作
 - A) Petmatch2(JavaScript)
 - B) Petmatch(Python)
 - C) Hlgame(クローン)
 - D) SmartHome

ページ

4
5
6
7

ページ

9
10
12
19
21
23

抜粋版

ページと本文は一致しません

24
31
36
40
45
48
48
49
54
55
56
61
62
64

Alexa開発キット 目次

Alexa開発

⑧ Alexa Voice Service開発

- 概要、用途
- AVS製品設定
- マイク、スピーカ設定
- AVS sample APP設定
- AVS sample APP動作
- AVS sample APP試験
- AVS sample APP LED試験
- AVS sample APPデバッグ
- デバイス確認
- Wakeup word設定(Snowboy)

ページ

[79](#)

[79](#)

[80](#)

[84](#)

[88](#)

[89](#)

[92](#)

[93](#)

[94](#)

[95](#)

[96](#)

抜粋版

ページと本文は一致しません



Pi運用マニュアル

1. Raspberry Piについて

既に全世界で1000万台以上販売された手のひらサイズのコンピュータです。
LinuxベースのRasbianOSで動作しております。

2. Linux基本コマンド

① システム関係

- 起動: 電源を入れると自動で起動します。
- 再起動: # reboot
又は、アプリケーション>ログアウト>再起動; 左上のメニューから
- 終了: # shutdown
又は、アプリケーション>ログアウト>シャットダウン; 左上のメニューから
- ログアウト # logout
又は、アプリケーション>ログアウト>ログアウト; 左上のメニューから
- **日本語／英語の入力切替**: キーボードのctl+jを同時に押します。又は右上のアイコン(右から7個目)からプルダウンで選択



Raspberry Pi

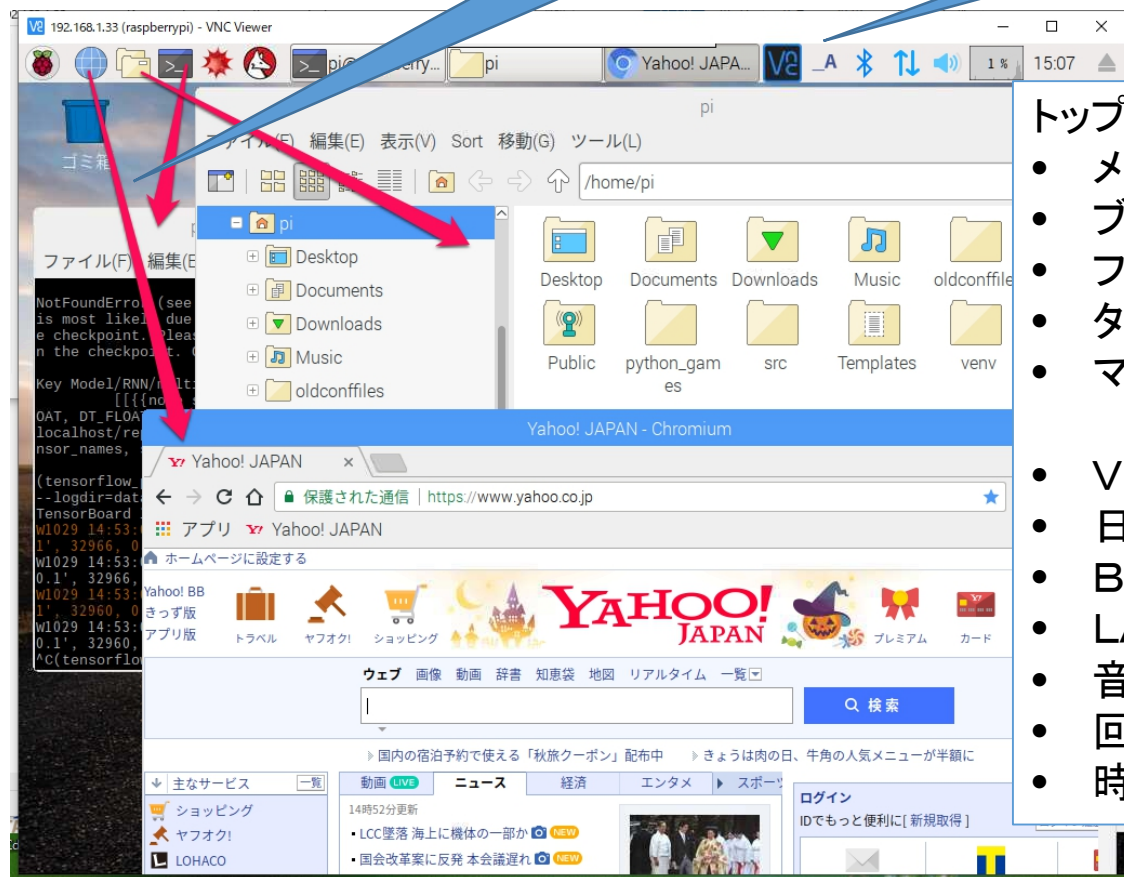
Pi運用マニュアル

3. 基本操作

① 表示画面と内容

トップ画面によく使うコマンド.txtをアップしてます

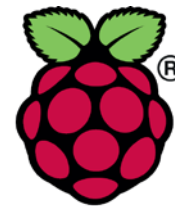
Anthonyが出ない場合は、一度Japaneseを選択後、再度Anthonyを選択してください。



トップ画面(上段のタスクバーで選択)

- メニュー
- ブラウザ
- ファイルマネージャ
- ターミナル
- マルチ画面選択

- VNC
- 日本語入力
- BLE
- LAN/WiFi
- 音量
- 回線効率
- 時刻



Pi運用マニュアル

4. 日常運用

① セキュリティ対策(アンチウイルス更新、スキャン)

- アンチウイルス対策として無料のclamAVをインストールしてます。
- 手動での運用を基本としています。

```
pi@raspberrypi: ~  
ファイル(F) 編集(E) タブ(T) ヘルプ(H)  
ERROR: /var/log/clamav/freshclam.log is locked by another  
ERROR: Problem with internal logger (UpdateLogFile = /var/  
og).  
root@raspberrypi: ~# leafpad /etc/clamav/freshclam.conf  
root@raspberrypi: ~# freshclam  
ClamAV update process started at Fri J  
main.cvd is up to date (version: 57, sigs: 4216790, f-level: 60, builder: mishh  
ammer)  
daily.cvd is up to date (version: 21862, sigs: 394456, f-level: 63, builder: neo  
)  
bytecode.cvd is up to date (version: 283, sigs: 53, f-level: 63, builder: neo)  
root@raspberrypi: ~# clamscan --infected --remove --recursive  
SCAN SUMMARY  
Known viruses: 4607906  
Engine version: 0.99.2  
Scanned directories: 264  
Scanned files: 2063  
Infected files: 0  
Data scanned: 61.31 MB  
Data read: 49.02 MB (ratio 1.25:1)  
Time: 71.844 sec (1 m 11 s)  
root@raspberrypi: ~#
```

パターンファイル更新

手動スキャン時に更新されます

手動でスキャン

clamscan --infected --remove --recursive

自動化可能ですが、バックグラウンドで重くなる可能性大。コマンド入力後約40分位かかります。

手動でスキャン

② インストール済パッケージの更新リスト、アップグレード

- Linuxの場合は、頻繁に更新が発生します。アップグレードを定期的 to 実施してください。
- 更新前には、バックアップを取ることをお勧めします。特にアップグレードはまれに動作不良、戻せない状態が発生します。自己責任で実施してください。

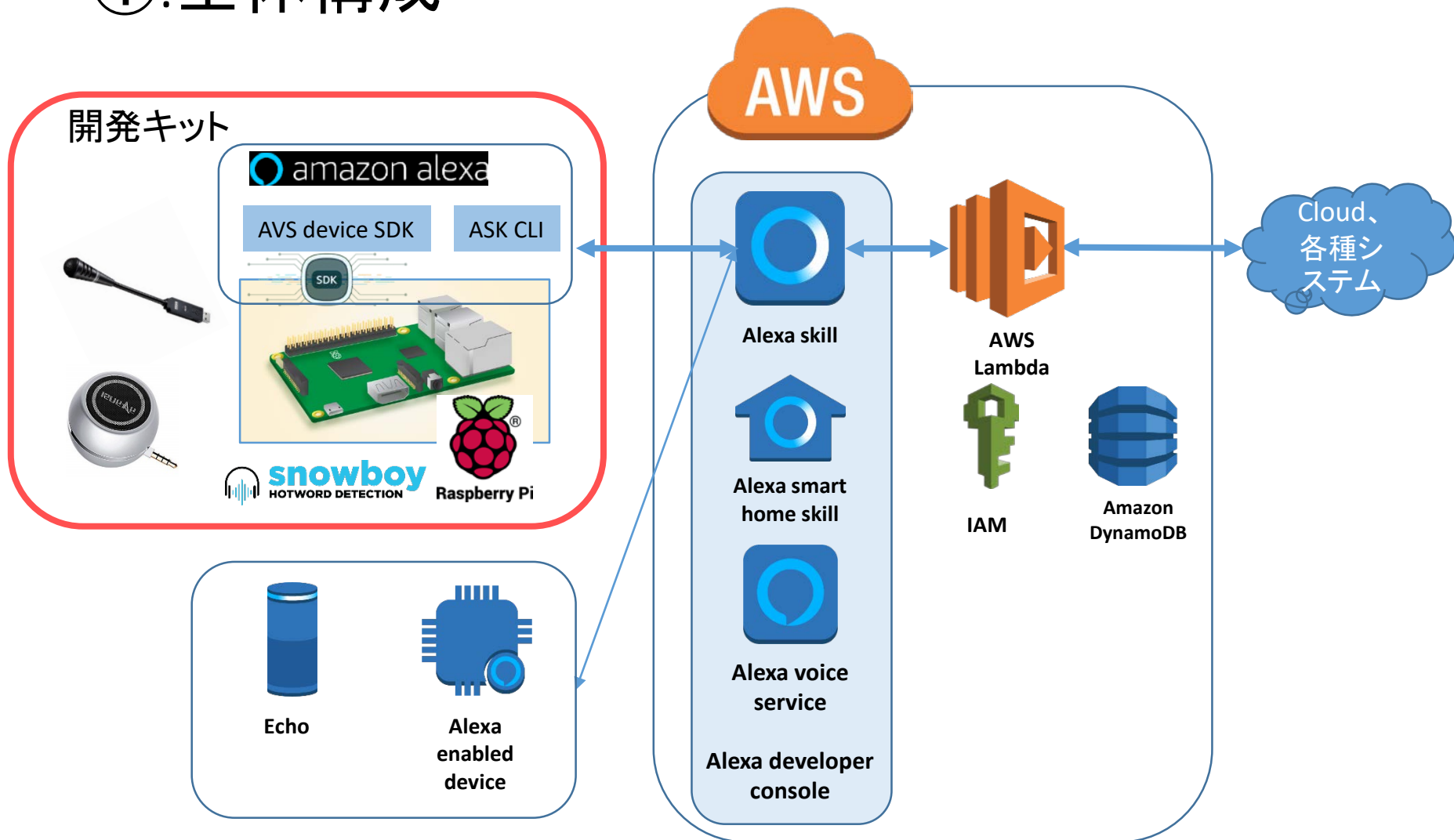
```
pi@raspberrypi: ~  
ファイル(F) 編集(E) タブ(T) ヘルプ(H)  
Mem: 925 596 329 9 28 381  
-/+ buffers/cache: 186 739  
Swap: 99 0 99  
root@raspberrypi: ~# apt-get update  
ヒット http://archive.raspberrypi.org jessie/contrib Translation-ja_JP  
ヒット http://mirrordirector.raspbian.org jessie/contrib Translation-ja_JP  
ヒット http://mirrordirector.raspbian.org jessie/contrib Translation-en  
ヒット http://archive.raspberrypi.org jessie/main Translation-ja_JP  
ヒット http://mirrordirector.raspbian.org jessie/main Translation-ja  
ヒット http://mirrordirector.raspbian.org jessie/main Translation-en  
ヒット http://mirrordirector.raspbian.org jessie/non-free Translation-ja_JP  
ヒット http://mirrordirector.raspbian.org jessie/non-free Translation-ja  
無視 http://archive.raspberrypi.org jessie/non-free Translation-en  
無視 http://archive.raspberrypi.org jessie/rpi Translation-en  
無視 http://archive.raspberrypi.org jessie/rpi Translation-en  
root@raspberrypi: ~# apt-get upgrade  
root@raspberrypi: ~#
```

```
更新リスト取得
# apt-get update
アップグレード実施
# apt-get upgrade
```

必ず実施前にバックアップ

Alexa開発

①.全体構成



Alexa開発

②. メニュー

- AWS設定

AWS Lambdaを使用するためのAWSの設定
AWSの詳細説明は以下のURLへ

https://aws.amazon.com/jp/cloud/?nc2=h_l2_cc

- Amazon. co. jp設定

Echoなどの端末を使用するための設定

<https://alexa.amazon.co.jp/>

- Amazon developer設定

Alexa developer console(Alexa skill kit, Alexa voice service)を使用するための設定。

<https://developer.amazon.com/ja/>

- Alexa開発(コンソール編)

- A) Hello world

アレクサスキル、Lambdaの導通確認ができます。

- B) Drink(スロット対応)

スロットを使用した会話例です。スロット部分に複数の選択肢を設定できます。非常によく使います。

- C) APL(画面对応)

Alexa Presentation Languageを使い、画面付きのAIスピーカ(Echo spot)に対応した会話例です。今後増加すると思われます。

- D) Device address

端末(Echoなど)の住所を問い合わせる会話例

- E) High Low game

0-100までの数字を当てるゲーム。コンピュータが想定の数値より上か、下かを回答し、正解まで繰り返します

Alexa公式サイト

<https://developer.amazon.com/ja/alexa-skills-kit>

詳細ドキュメント

<https://developer.amazon.com/ja/docs/ask-overviews/build-skills-with-the-alexa-skills-kit.html>

Alexa開発

②. メニュー

Alexa開発 (CLI編)

- 概要、特徴
 - 認証設定の準備: IAM設定
 - 認証設定: .aws, .ask設定
 - 基本操作
- A) Petmatch2 (JavaScript): 好きなペットを選択するスキルのJS版
 - B) Petmatch (Python): 上記のpython版
 - C) Hlgame (クローン): ハイアンドローゲーム、コンソール編のクローン
 - D) SmartHome: スマートホームAPI

Alexa Voice Service開発

- 概要、用途
- AVS製品設定: Alexa端末として製品登録します
- マイク、スピーカ設定: Piに接続するデバイスの設定
- AVS sample APP設定: アマゾンが提供するサンプルアプリです。
- AVS sample APP動作
- AVS sample APP試験
- AVS sample APP LED試験
- AVS sample APPデバッグ
- デバイス確認: 登録したデバイスの確認
- Wakeup word設定 (Snowboy): Alexa以外の起動ワードを作成、設定します。

Alexa公式サイト

<https://developer.amazon.com/ja/alexa-skills-kit>

詳細ドキュメント

<https://developer.amazon.com/ja/docs/ask-overviews/build-skills-with-the-alexa-skills-kit.html>

Alexa開発

③. AWS設定

• 手順1: アカウントの作成

- <https://aws.amazon.com/jp/register-flow/>
- 必要なメールアドレス、パスワードなどを入力します。

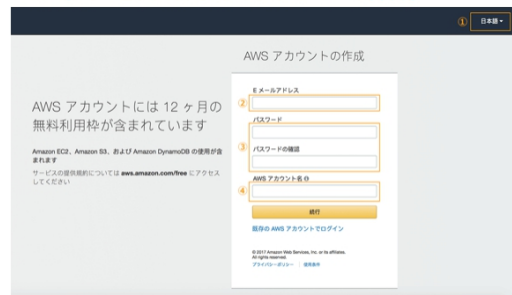


AWS アカウント作成の流れ

AWS アカウントを作成すると、1年間の無料利用枠はもちろん、AWS クラウドの世界中のリージョンで提供されるすべてのサービスを始めることができます。こちらでは日本のお客様に AWS アカウント作成におけるポイントをご紹介します。



ステップ 1: AWS アカウントの作成



※クリックすると大きな画像でご覧いただけます。

このページの上部タイトルおよび、末尾に設置されているオレンジ色のアカウント作成ボタンよりサインアップ画面へ移動します。

各ページ右上 ① の言語選択ボックスより、「日本語」でない場合「日本語」を選択後、こちらのサインアップ画面へお進みください。

最初に AWS アカウントとなる情報を設定します。

- ② の「E メールアドレス」には、AWS へのログインに利用したいメールアドレスを設定します。(※)
- ③ の「パスワード」および「パスワードの確認」で AWS へのログイン時に使用するパスワードを設定し、さらに確認用にもう一度同じパスワードを入力します。
- ④ の「AWS アカウント名」テキストボックスに、お客様のお名前を半角アルファベットで入力します。
- 入力後、「続行」ボタンをクリックします。

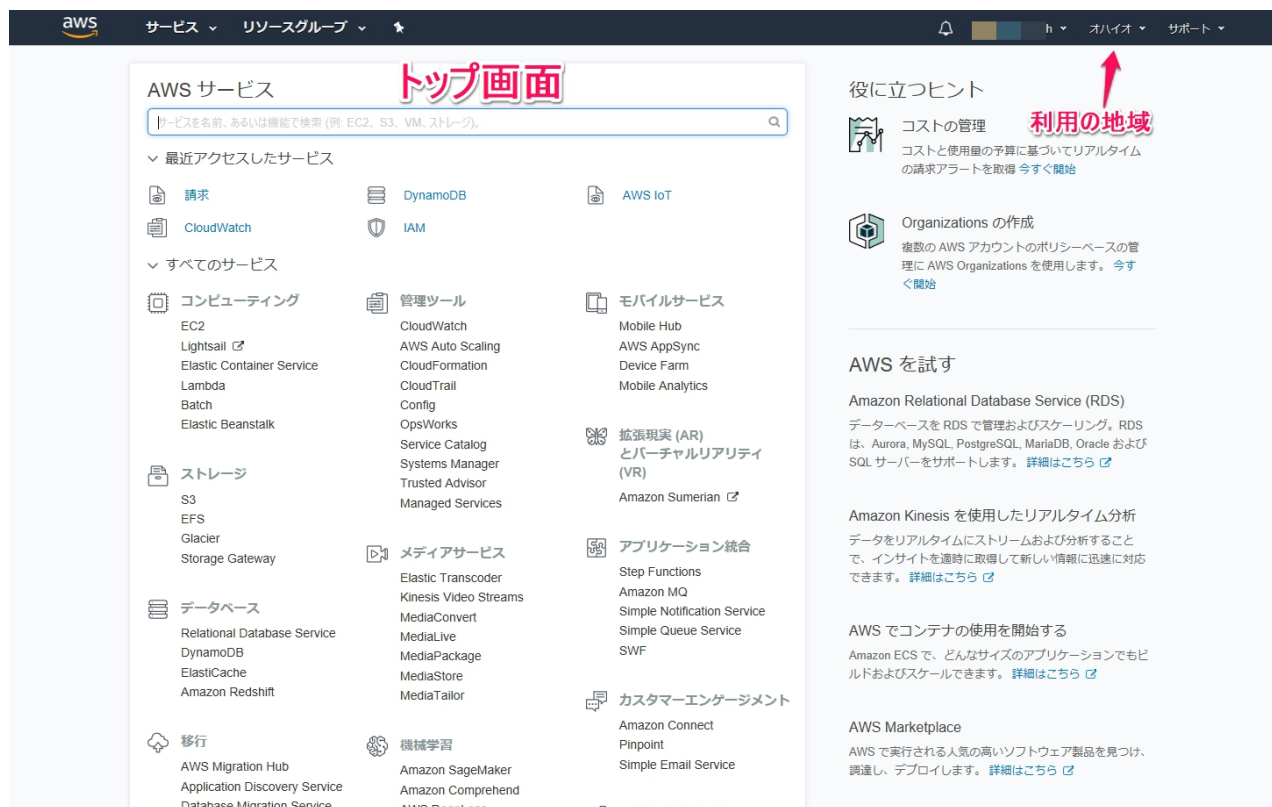
※ご登録いただくメールアドレスは、AWS 側からの通知等にも利用されます。複数の方へ

Alexa開発

③. AWS設定

• AWSトップ画面

- 利用する場合に、地域を意識して設定してください。地域毎に料金が変わったり、利用できるサービスが限定されている場合があります。



Alexa開発

④. Amazon. co. jp設定



Echo

- アカウント作成
 - Echoなどの設定をするためにAmazon. co. jpのアカウントを作成
<https://www.amazon.co.jp/>
 - このアカウントで、Amazon developerでも同じものを使用します。

amazon.co.jp

アカウントを作成

名前

フリガナ

Eメールアドレス

パスワード

最低6文字必要です
パスワードの長さは最低6文字です。

もう一度パスワードを入力してください

Amazonアカウントを作成

ログインすることにより、当社の利用規約およびプライバシー規約に同意したとみなされます。

すでにアカウントをお持ちですか? ログイン

利用規約 プライバシー規約 ヘルプ

© 1996-2019, Amazon.com, Inc. or its affiliates



⑤. Amazon developer設定



Alexa skill

- Alexa skill kitを選択
 - Amazon developerからAlexa skill kitを選択します。

<https://developer.amazon.com/alexa/console/ask>

alexa developer console

フィードバックフォーラム

Alexaスキルで収益につなげましょう（現在このプログラムは日本のスキルには適用されません）
(米国向けスキルでのみ有効) スキル内課金でプレミアムコンテンツやサブスクリプションを販売して、Alexaスキルのエクスペリエンスを向上させましょう。特典や価格を設定すれば、音声ファーストの課金プロセスをご利用できます。プレミアムスキルの開発には、ASKコマンドラインインターフェースを使用する必要があります。CLIの使用法についてはこちら。

Alexa skill kit

Alexa Skills Kit開発者コンソールにようこそ

新機能や新規ツールについての詳細はリリースノートをご覧ください。このビデオを見るや技術資料を読んでくださいで最新情報を入手しましょう。

スキル 収益 支払い

Alexaスキル

スキルの作成

スキル名	言語	タイプ	更新日	ステータス	アクション
smarthome_jp スキル ID の表示	日本語(日本)	スマートホーム	2019-05-09	● 開発中	レポート 編集 ベータテストの管理 削除
My Radio スキル ID の表示	イタリア語(イタリア), 英語(米国), 英語(カナダ), 英語(インド), 英語(オーストラリア), スペイン語(スペイン), フランス語(フランス), 日本語(日本), 英語(英国), フランス語(カナダ), スペイン語(メキシコ)	カスタム	2019-05-08	● 開発中	レポート 編集 削除

開発、公開中のスキル一覧

Alexa開発

⑥. Alexaスキル開発(コンソール編)



Alexa skill

- Alexa developer consoleから開発

<https://developer.amazon.com/alexa/console/ask>

- パソコンの画面で上記にアクセス、Raspberry PiはVNC接続しておきます。
- ビルド、コードエディタ、テストを使いコンソールのみで簡単な開発が可能

The screenshot displays the Alexa Developer Console interface. On the left, the 'Intent' list includes 'PetMatchIntent' with slots for 'pet', 'size', 'temperament', 'I_Want', 'energy', 'article', 'comparison', 'amount', 'units', 'shedding', 'location', and 'at_the'. The main area shows the 'Alexa シミュレータ' (Alexa Simulator) with a sample utterance: 'Welcome to pet match. I can help you find the best dog for you. What are two things you are looking for in a dog?'. The right panel shows the 'JSON出力' (JSON Output) for the skill invocation, including session information, user ID, device ID, and the skill's API endpoint.

Alexa開発

⑥. Alexaスキル開発(コンソール編)



Alexa skill

A) Hello world

<https://developer.amazon.com/alexa/console/ask>

- スキル作成
- スキル名、カスタム、Alexaがホスト、スキル作成を入力

alexa developer console

Alexaスキルで収益につなげましょう (現在このプログラムは日本のスキルには適用されません)

(米国向けスキルでのみ有効) スキル内課金でプレミアムコンテンツやサブスクリプションを販売して、Alexaスキルのエクスペリエンスを向上させましょう。特典や価格を設定すれば、音声ファーストの課金プロセスをご利用できます。プレミアムスキルの開発には、ASKコマンドラインインターフェースを使用する必要があります。CLIの使用方法についてはこちら。

Alexa Skills Kit開発者コンソールにようこそ

新機能や新ツールについての詳細はリリースノートをご覧ください。このビデオを見るや技術資料を読んでください最新情報を入手しましょう。

スキル 収益 支払い

Alexaスキル

スキル名	言語	タイプ	更新日	ステータス	アクション
smarthome_jp スキル ID の表示	日本語(日本)	スマートホーム	2019-05-09	開発中	レポート 編集 ベータテストの管理 削除
My Radio スキル ID の表示	イタリア語(イタリア), 英語(米国), 英語(カナダ), 英語(インド), 英語(オーストラリア), スペイン語(スペイン), フランス語(フランス), 日本語(日本), 英語(英国), フランス語(カナダ), スペイン語(メキシコ)	カスタム	2019-05-08	開発中	レポート 編集 削除

スキルの作成

新しいスキルを作成

キャンセル スキルを作成

スキル名
hello_world

スキル名は25文字以内で入力する必要があります。

デフォルトの言語
日本語(日本)

有償版に課金を設定することもできます。

スキルに追加するモデルを選択

自分たちのカスタムモデルを設計したり、プリビルドモデルを使用したりなど、さまざまな方法でスキルの内蔵を定められます。プリビルドモデルとは、インデントと英語のバリエーションを含み、スキルに追加できる別モデルです。

カスタム
ユニークなスキルを作成しましょう。カスタムモデルでスキルの対応をすべて作成できます。

クラッシュブリーフィング
ユーザーが自分のニュースフィードを管理できます。このアプリビルドモデルを使用すると、開きたいコンテンツを自分で管理できるほか、ジャンル別にトピックを探索できます。

スマートホーム
ユーザーが自分のスマートホームデバイスを管理できます。このプリビルドモデルを使用すると、開きたい他のデバイスの電源をオン/オフにできます。

「アレクサ、クラッシュブリーフィングを教えてください」

「アレクサ、キッチンライトをオンにして。」

スキルのバックエンドリソースをホスティングする方法を選択

ユーザーのバックエンドリソースをプロビジョニングすることも、Alexaにホストさせることもできます。Alexaにホストさせる場合は、コードになります。コードエディターを使用すると、開発者コンソールから直接AWS Lambdaにコードをデプロイできます。

独自のプロビジョニング
ユーザーのスキルエンドポイントとバックエンドリソースをプロビジョニングします。このオプションを使用すると、コンソールのコードエディターを使用できません。

Alexaがホスト
AlexaはAWSの無料上層でユーザーのアカウントのスキルをホスティングします。AWS Lambdaエンドポイント、メディアストレージS3、毎月のデータ転送量15GB、セッション永続性用のテーブルにアクセスできます。詳しくはこちら

ホストを選ぶと、lambda関数は、awsサービスで表示されません

Alexa開発

⑥. Alexaスキル開発(コンソール編)



Alexa skill

A) Hello world

<https://developer.amazon.com/alexa/console/ask>

- ビルト>カスタム
- Jsonエディタ>ファイルをドラッグ & ドロップ
- モデル保存、モデルをビルト

Windowsネットワークから
RaspberryPi をクリックして行きます。
でない場合はIPアドレスを入力

Raspberry Piのフォルダから
/home/pi/alexa/ask_con/helloworld
hello_world.jsonをドラッグ & ドロップ

alexa developer console

開発を開始するには

スキルビルダー

JSONエディター

1. 呼び出し名 > スキルの呼び出し名を

2. インテント、サンプル > 少なくとも1つのインテントを追加してください

3. モデルをビルド > 正常に対話モデルをビルド

4. エンドポイント > ウェブサービスのエンドポイントのリンクを追加

スキル内商品 > スキル内商品を作成

JSONエディター

対話モデルのスキーマ定義の詳細については、[ここをクリックしてください](#)。

jsonファイルをドラッグ&ドロップ

```

{
  "interactionModel": {
    "languageCode": "ja",
    "invocationName": "ハローワールド",
    "intents": [
      {
        "name": "AMAZON.CancelIntent",
        "samples": []
      },
      {
        "name": "AMAZON.HelpIntent",
        "samples": []
      },
      {
        "name": "AMAZON.StopIntent",
        "samples": []
      },
      {
        "name": "HelloWorldIntent",
        "slots": [],
        "samples": [
          "はいと書いて",
          "ハロー"
        ]
      },
      {
        "name": "AMAZON.NavigateHomeIntent",
        "samples": []
      }
    ],
    "types": []
  }
}

```

⑥. Alexaスキル開発(コンソール編)



Alexa skill

A) Hello world

<https://developer.amazon.com/alexa/console/ask>

- ビルト>カスタム
- 呼出し名確認: ハローワールド: Echoからスキルを呼び出す時の名前
- インテント: HelloWorldIntent: 会話のサンプルを入力します

カスタム

対話モデル

呼び出し名

インテント(5)

HelloWorldIntent

ビルトインインテント(4)

AMAZON.CancelIntent

AMAZON.HelpIntent

AMAZON.StopIntent

AMAZON.NavigateHomeIntent

スロットタイプ(0)

JSONエディター

インターフェース

エンドポイント

インテント履歴

画面表示

スキル内商品

アカウントリンク

アクセス権限

呼び出し名

特定のカスタムスキルとの対話を開始するには、スキルの呼び出し名を発話します。たとえば、呼び出し名が「今日の星占い」の場合、次のように話しかけます。

ユーザー:アレクサ、今日の星占いを聞いて双子座の運勢を占って

スキルの呼び出し名

ハローワールド

呼び出し名の要件

呼び出し名は2語以上でなければなりません。また、使用できるのはひらがな、カタカナ、漢字のみです。数字などは文字で表現しなければなりません。(例:「二十」など)

呼び出し名には、「起動して」、「聞いて」、「教えて」、「読み込んで」、「起動プレースを使用することはできません。「アレクサ」、「アマゾン」、「エコ」、「コンピ」、「スキル」、「アプリ」などの単語は使用できません。カスタムスキルの呼び出し名についての詳細はここらへ。

スキルの対話モデルをビルドするまで、スキルの呼び出し名の変更は反映されません。ビルドを話モデルに少なくとも1つのサンプル発話のあるインテントが含まれている必要があります。カについての詳細はここらへ。

alexa developer console

日本語

モデルを保存

モデルをビルド

インテント / HelloWorldIntent

サンプル発話(2)

追加する場合

はいと言って

ハロー

ダイアログデリゲートのルール

このインテントでは、ダイアログ管理...

これが無効にされている理由

インテントスロット(0)

順序	名前	スロットタイプ	アクション
1	新しいスロットを作成	スロットタイプを選択	ダイアログを編集 削除

スロットタイプを選択

ダイアログを編集 | 削除

インテントの確認

© 2010 - 2019, Amazon.com, Inc. or its affiliates. All Rights Reserved. 無断複製 転載を禁じます 利用規約 文書 フォーラム Alexaブログ 開発者ポータルホーム

⑥. Alexaスキル開発(コンソール編)

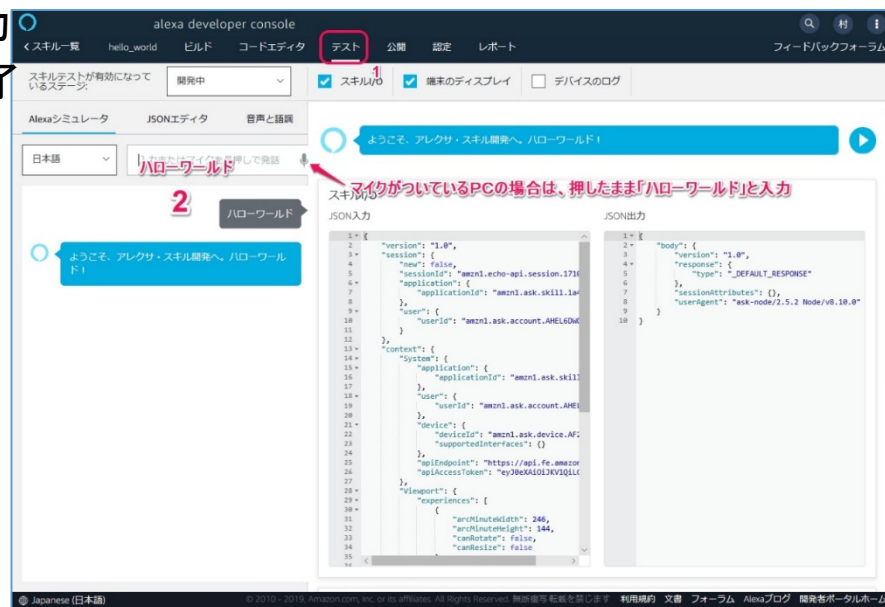


Alexa skill

A) Hello world

<https://developer.amazon.com/alexa/console/ask>

- テスト 作成したスキルのテストを画面上で行います。
- 開発中になっていることを確認します。
- ハローワールドと入力します。
- ようこそ、アレクサ・スキル開発・
- また、マイクがついている場合は声で入力
- 正常動作の場合は、Json出力されて完了

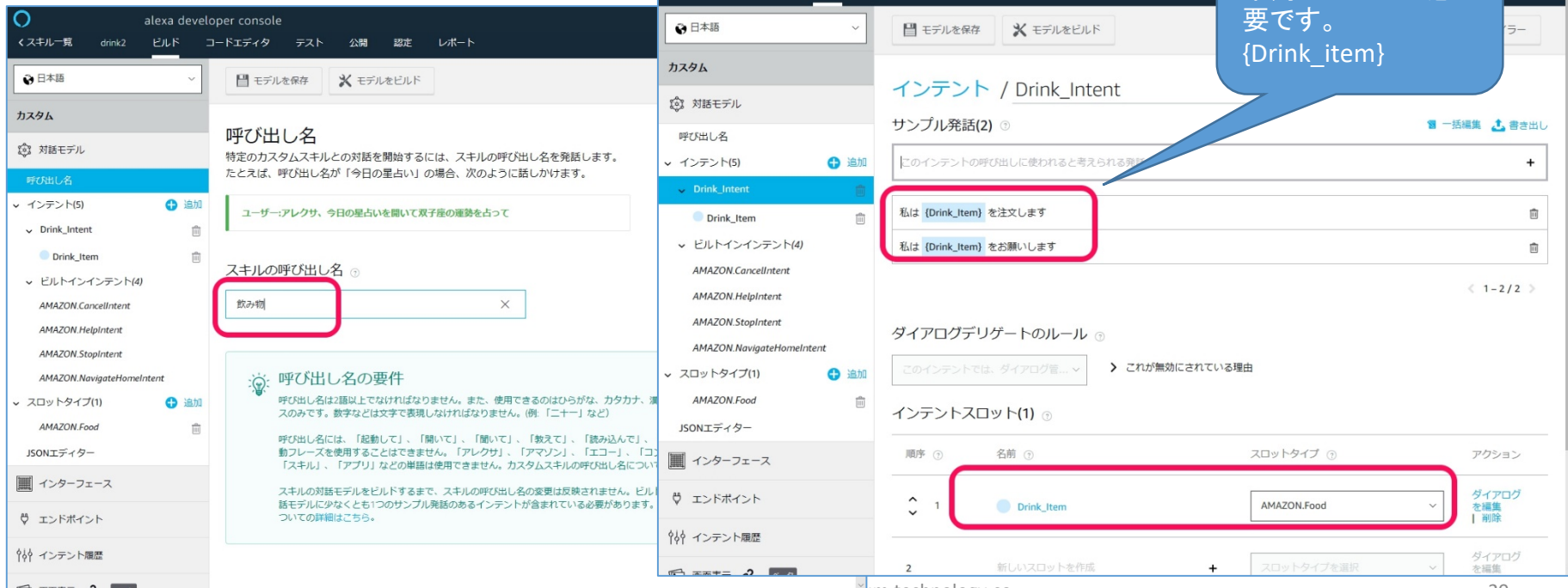


⑥. Alexaスキル開発(コンソール編)

B) Drink スロット対応

- スロットを使用した会話例です。複数の選択肢を用意します。非常によく使います。
- ビルト>カスタム
- 呼出し名確認: 飲み物: Echoからスキルを呼び出す時の名前
- インテント: Drink_Intent: 会話のサンプルを入力します
- Drink_item: スロットを設定

スロットの前後には
半角スペースが必要
です。
{Drink_item}



alex developer console

日本語

カスタム

対話モデル

呼び出し名

特定のカスタムスキルとの対話を開始するには、スキルの呼び出し名を登録します。たとえば、呼び出し名が「今日の星占い」の場合、次のように話しかけます。

ユーザー:アレクサ、今日の星占いを聞いて双子座の運勢を占って

スキルの呼び出し名

飲み物

呼び出し名の要件

呼び出し名は2語以上でなければなりません。また、使用できるのはひらがな、カタカナ、数字のみです。数字などは文字で表現しなければなりません。(例「二十一」など)

呼び出し名には、「起動して」、「聞いて」、「教えて」、「読み込んで」、「動プレースを使用することはできません。「アレクサ」、「アマゾン」、「エコー」、「コン」、「スキル」、「アプリ」などの単語は使用できません。カスタムスキルの呼び出し名については詳細はこちら。

スキルをビルドするまで、スキルの呼び出し名の変更は反映されません。ビルド完了後、スキルの呼び出し名が変更されている必要があります。

intent / Drink_Intent

サンプル発話(2)

このIntentの呼び出しに使われると考えられる発話

私は {Drink_Item} を注文します

私は {Drink_Item} をお願いします

ダイアログデリゲートのルール

このIntentでは、ダイアログ管理... > これが無効にされている理由

Intentスロット(1)

順序	名前	スロットタイプ	アクション
1	Drink_Item	AMAZON.Food	ダイアログを編集 削除

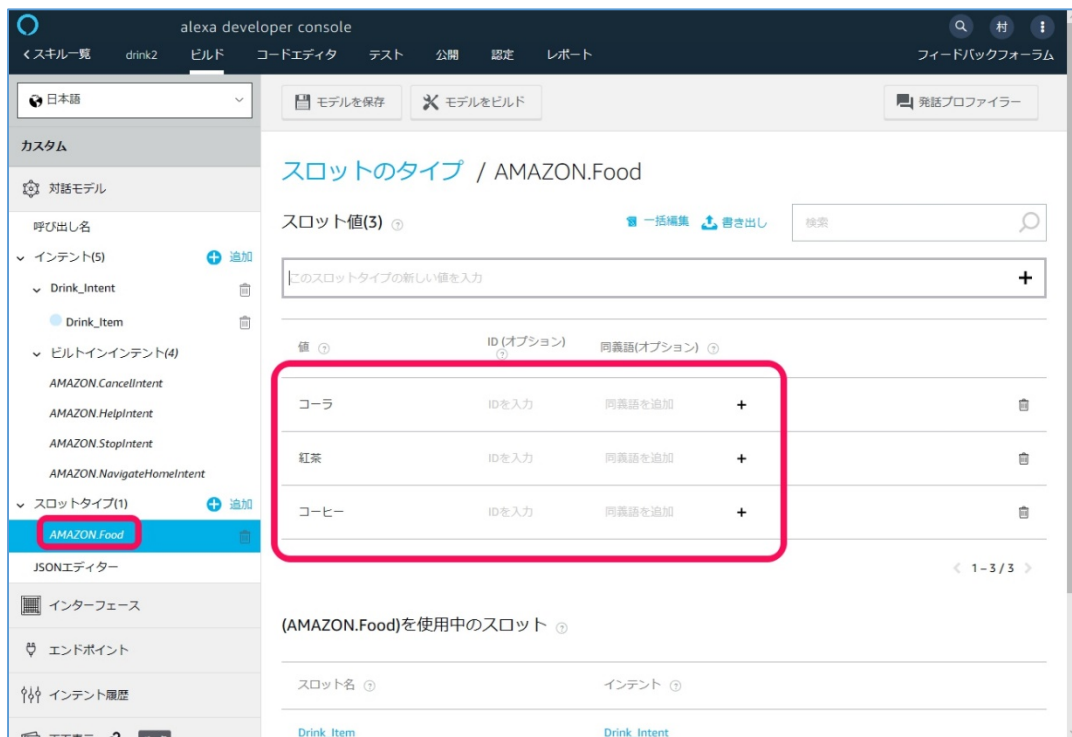


⑥. Alexaスキル開発(コンソール編)

B) Drink スロット対応

<https://developer.amazon.com/ja/docs/custom-skills/create-intents-utterances-and-slots.html>

- 上記のマニュアルをよく理解しましょう。
- ビルト>スロットタイプ スロット・タイプには、標準とカスタムがあります
- Amazon. Food>スロットに値を入力



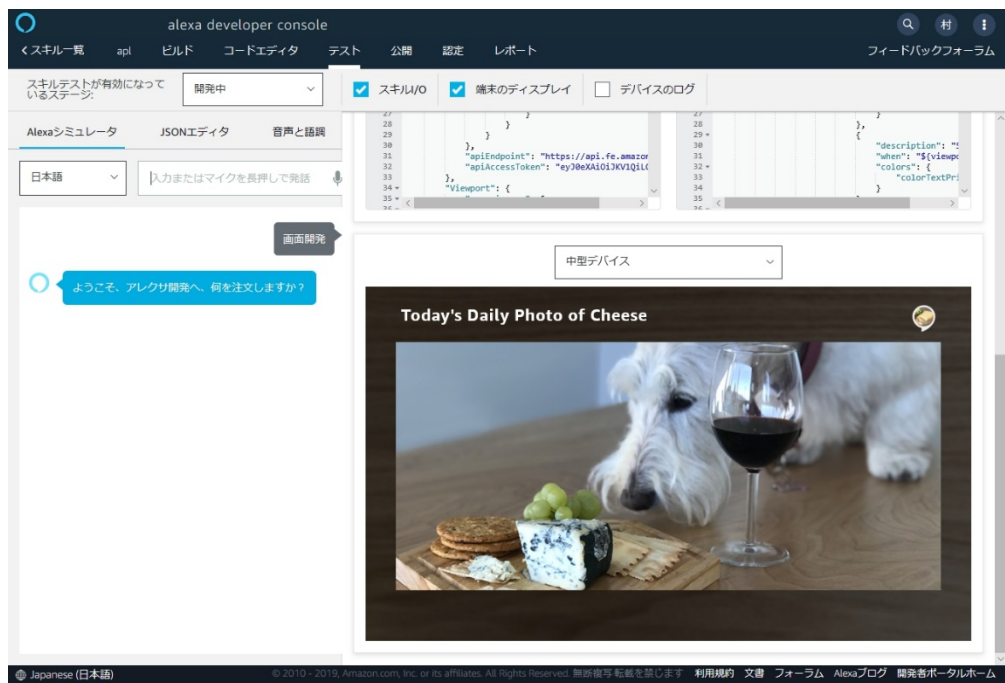


⑥. Alexaスキル開発(コンソール編)

C) APL 画面対応

<https://developer.amazon.com/ja/blogs/alexa/post/a3712152-cd02-4fda-9d33-3b64ef3a13af/jp-alexa-presentation-language-part1>

- テスト 作成したスキルのテストを画面上で行います。
- 開発中になっていることを確認します。
- “画面開発”と入力します。
- 下の方に画面が現れます。



Echo spotでの表示例



⑥. Alexaスキル開発(コンソール編)



Alexa skill

D) Device Address

- 使用しているデバイスの住所を回答する例
- スキル作成、addressで入力します
- ビルト>カスタム
- Jsonエディタ>ファイルをドラッグ & ドロップ
- モデル保存、モデルをビルド

Raspberry Piのフォルダから
/home/pi/alexa/ask_con/device_address
address.jsonをドラッグ & ドロップ

⑥. Alexaスキル開発(コンソール編)



Alexa skill

D) Device Address

- テスト 作成したスキルのテストを画面上で行います。
- 開発中になっていることを確認します。
- “装置アドレス”と入力します。
- “住所は”と入力します。
- 端末の住所を回答

alex developer console

スキル一覧 address ビルド コードエディタ **テスト** 公開 認定 レポート

スキルテストが有効になっているステージ: 開発中

☒ スキル/O ☒ 端末のディスプレイ ☐ デバイスのログ

Alexaシミュレータ JSONエディタ 音声と語調

日本語 入力またはマイクを長押しして発話

装置アドレス

住所は

あなたのアドレスは、所沢市御幸町, 埼玉県 359-1115

スキル/O

JSON入力

```
1 {
2   "version": "1.0",
3   "session": {
4     "new": false,
5     "sessionId": "amzn1.echo-api.session.272d",
6     "applicationId": "amzn1.ask.skill.6c",
7     "user": {
8       "userId": "amzn1.ask.account.AEP63R2",
9       "permissions": {
10        "consentToken": "eyJ0eXAiOiJKV1QiLC
11      }
12    }
13  },
14  "context": {
15    "System": {
16      "applicationId": "amzn1.ask.skill",
17      "user": {
18        "userId": "amzn1.ask.account.AEP6",
19        "permissions": {
20          "consentToken": "eyJ0eXAiOiJKV1QiLC
21      }
22    }
23  },
24  "device": {
25    "deviceId": "amzn1.ask.device.AHP",
26    "supportedInterfaces": {}
27  },
28  "apiEndpoint": "https://api.fe.amazon",
29  "apiAccessToken": "eyJ0eXAiOiJKV1QiLC
30  },
31  }
```

JSON出力

```
1 {
2   "body": {
3     "version": "1.0",
4     "response": {
5       "outputSpeech": {
6         "type": "SSML",
7         "ssml": "<cspeak>あなたのアドレスは、",
8         "type": "_DEFAULT_RESPONSE"
9       },
10      "sessionAttributes": {},
11      "userAgent": "ask-node/2.5.2 Node/v8.10.0"
12    }
13  }
```

© Japanese (日本語)

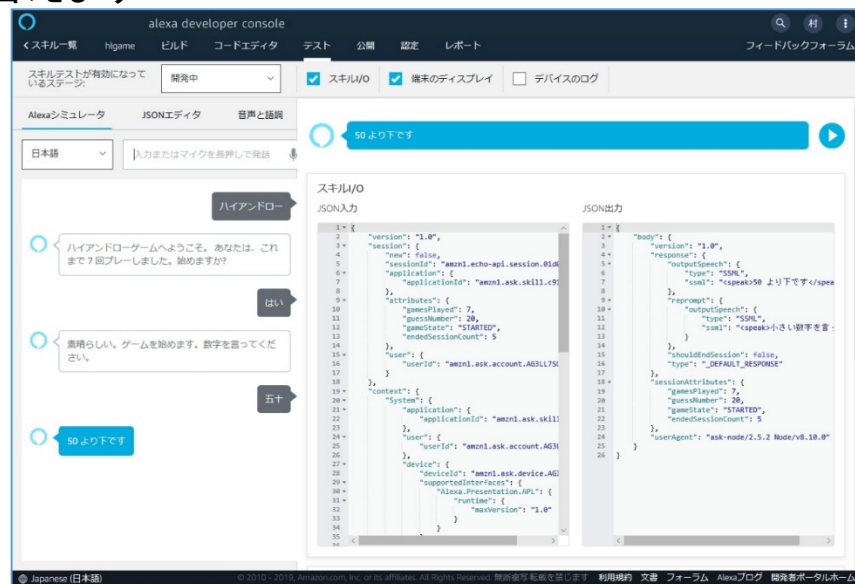
© 2010 - 2019, Amazon.com, Inc. or its affiliates. All Rights Reserved. 無断複写転載を禁じます 利用規約 文書 フォーラム Alexaブログ 開発者ポータルホーム



⑥. Alexaスキル開発(コンソール編)

E) High low game

- ハイアンドローゲーム。0ー100までの数字を言って、数字を当てるゲームです。
- テスト 作成したスキルのテストを画面上で行います。
- 開発中になっていることを確認します。
- “ハイアンドロー”と入力します。
- “はい”と入力します。
- 数字(英数字はNG)を入力
- コンピュータが設定した数字より上か、下を答えます
- どんどん数字を入力し、正解を当てます



⑦. Alexaスキル開発(CLI編)



Alexa skill

• 概要

- Alexa skill kitをRaspberry Piを使って、CLI(Command Line interface)で動作させます。詳細は、以下のドキュメント参照。
- <https://developer.amazon.com/ja/docs/smapi/quick-start-alexa-skills-kit-command-line-interface.html>

• 特徴

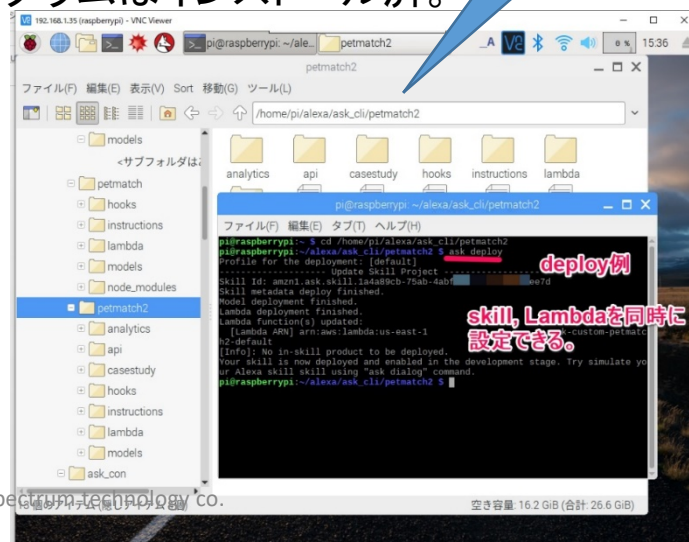
- 動作が早い、マルチリージョン、多言語対応
- スキル開発、Lambda開発、テストを一貫して実施できる。
- Javascriptのプログラム知識が必要
- 認証関係の設定を行うだけ。必要なプログラムはインストール済。

トップ画面によく使うコマンド.txtをアップしてます

```
pi@raspberrypi: ~/alexa/ask_cli/petmatch2
ファイル(F) 編集(E) タブ(T) ヘルプ(H)

pi@raspberrypi:~$ cd /home/pi/alexa/ask_cli/petmatch2
pi@raspberrypi:~/alexa/ask_cli/petmatch2$ ask deploy
Profile for the deployment: [default]
----- Update Skill Project -----
Skill Id: amzn1.ask.skill.1a4a89cb-7ee7d
Skill metadata deploy finished.
Model deployment finished.
Lambda deployment finished.
Lambda function(s) updated:
[Lambda ARN] arn:aws:lambda:us-east-1:495...ion:ask-custom-petmatch2-default
[Info]: No in-skill product to be deployed.
Your skill is now deployed and enabled in the development stage. Try simulate your Alexa skill using "ask dialog" command.
pi@raspberrypi:~/alexa/ask_cli/petmatch2$ ask dialog -l ja-JP
User > ペットマッチ
Alexa > Welcome to pet match. I can help you find the best dog for you. What are two things you are looking for in a dog?
User >
```

会話も可能



deploy例
skill, Lambdaを同時に
設定できる。



Alexa開発

⑦. Alexaスキル開発(CLI編)

- 基本操作

- ドキュメント

- <https://developer.amazon.com/ja/docs/smapi/manage-credentials-with-ask-cli.html>

- 新規作成

- \$ ask new

- \$ ask new --url <https://github.com/alexa/skill-sample-nodejs-petmatch.git>

githubからダウン

- ロードして作成

- デプロイ

- \$ ask deploy

新規にインストールしたフォルダに移動し、実施。skill.jsonがあると動作

失敗する場合は、IAMでユーザにlambdaにアクセスできるポリシー設定されているか確認

Skill-idは自動で付与

- Lambda確認

- デプロイが実施されたか確認、リージョンは、バージニア(us-east-1)がデフォルト

- <https://console.aws.amazon.com/lambda/home?region=us-east-1#/functions>

- スキル確認

- デプロイされたスキルを確認します。

- <https://developer.amazon.com/alexa/console/ask?>

- 会話試験

- \$ ask dialog -l ja-JP

ローカルを指定します。

- >user

呼出し名を入力します。

- シミュレーション

- \$ ask simulate -l ja-JP -t “ペットマッチ”

呼出し名を付加して入力すると、詳細のjson結果が出力

- スキル削除

- \$ ask api delete-skill -s **skill_id** スキルIDを入れて削除



⑦. Alexaスキル開発(CLI編)

A) Petmatch2 jsで作成したスキル

- 自分の好みのペットを選択して行きます。

\$ ask dialog -l ja-JP 会話の試験が可能

\$ ask simulate -l ja-JP -t “ペットマッチ”

シミュレーションが可能

Piのフォルダ位置

/home/pi/alexa/ask_cli/petmatch2

\$ ask dialog -l ja-JP

\$ ask simulate -l ja-JP -t “ペットマッチ”

```
pi@raspberrypi: ~/alexa/ask_cli/petmatch2
ファイル(F) 編集(E) タブ(T) ヘルプ(H)

models
├── en-US.json
├── ja-JP.json
└── skill.json

37 directories, 181 files
pi@raspberrypi:~/alexa/ask_cli/petmatch2 $ ask deploy
Profile for the deployment: [default]
----- Update Skill Project -----
Skill Id: amzn1.ask.skill.1a4a89cb-75ab-4abf-b323-88844b00ee7d
Skill metadata deploy finished.
Model deployment finished.
Lambda deployment finished.
Lambda function(s) updated:
[Lambda ARN] arn:aws:lambda:us-east-1:495029905165:function:ask-custom-h2-default
[Info]: No in-skill product to be deployed.
Your skill is now deployed and enabled in the development stage. Try simulating your Alexa skill using "ask dialog" command.
pi@raspberrypi:~/alexa/ask_cli/petmatch2 $ ask dialog -l ja-JP
User > ペットマッチ
Alexa > Welcome to pet match. I can help you find the best dog for you.
are two things you are looking for in a dog?
User >
```

```
pi@raspberrypi: ~/alexa/ask_cli/petmatch2
ファイル(F) 編集(E) タブ(T) ヘルプ(H)

are two things you are looking for in a dog?
User >
pi@raspberrypi:~/alexa/ask_cli/petmatch2 $ ask simulate -l ja-JP -t "ペットマッチ"
✓ Simulation created for simulation id: 09503cce-e98e-4ab9-...
{
  "id": "09503cce-e98e-4ab9-b2-...-5",
  "status": "FAILED",
  "result": {
    "alexaExecutionInfo": {
      "consideredIntents": [
        {
          "name": "PetMatchIntent",
          "confirmationStatus": "NONE",
          "slots": {
            "at_the": {
              "name": "at_the",
              "confirmationStatus": "NONE"
            },
            "amount": {
              "name": "amount",
              "confirmationStatus": "NONE"
            },
            "comparison": {
```

json出力

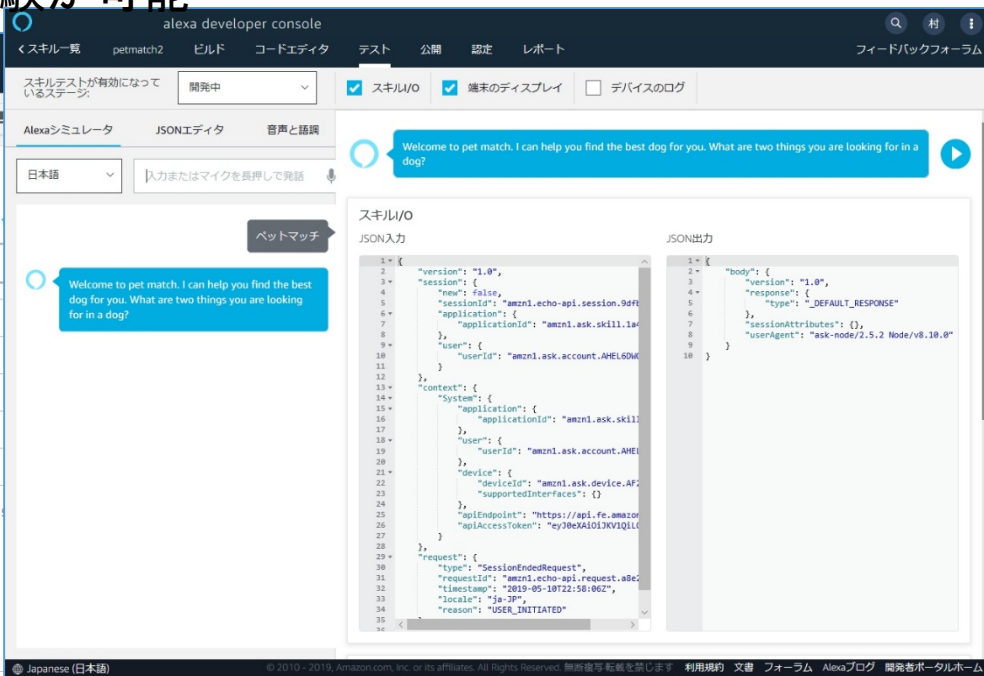
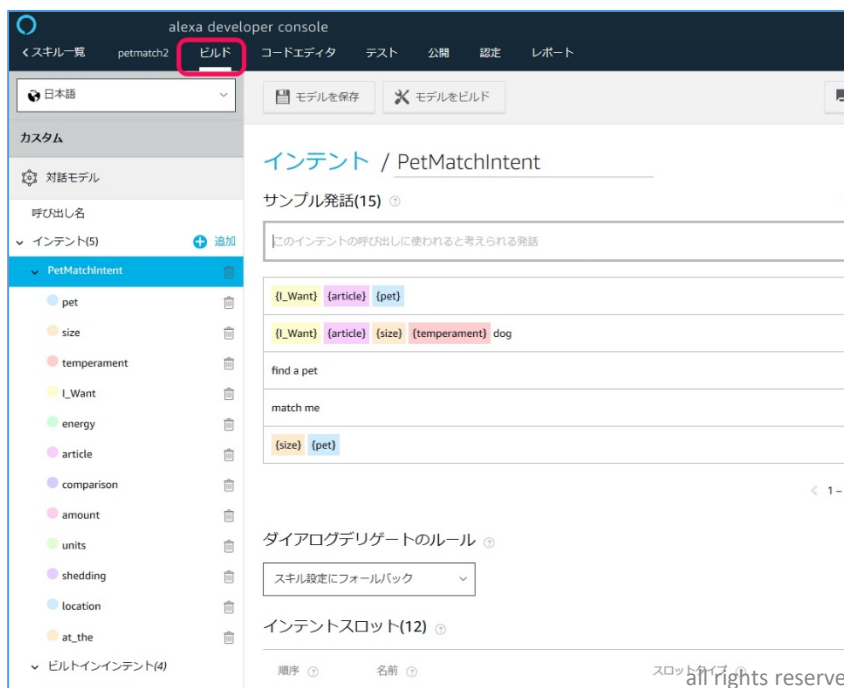
⑦. Alexaスキル開発(CLI編)



Alexa skill

A) Petmatch2 jsで作成したスキル

- 自分の好みのペットを選択して行きます。
- Alexa developer consoleでのテスト
- <https://developer.amazon.com/alexa/console/ask>
- petmatch2を選択し、ビルトで内容確認
- テスト>ペットマッチ入力して試験が可能





Alexa開発

⑦. Alexaスキル開発(CLI編)

C) Hlgame jsで作成したスキル

- ハイアンドローゲームのコンソールで開発したものをローカルにクローンとして作成

- 複数個所で作業する時に便利

\$ ask dialog -l ja-JP 会話のテストも可能

Piのフォルダ位置
/home/pi/alexa/ask_cli/hlgame
\$ ask dialog -l ja-JP

```
pi@raspberrypi: ~/alexa/ask_cli/hlgame
ファイル(F) 編集(E) タブ(T) ヘルプ(H)
\ No newline at end of file
-----
--- local models/ja-JP.json
+++ remote models/ja-JP.json
@@ -70,5 +70,5 @@
    "types": []
  }
},
"version": "10"
-}
+}
\ No newline at end of file

===== ALEXA HOSTED SKILL DIFF =====
You can use git diff commands as you like.
Example: "git diff origin/master" to see the diff between your remote dev skill
and your actual code
pi@raspberrypi:~/alexa/ask_cli/hlgame $ ask dialog -l ja-JP
User > ハイアンドロー
Alexa > ハイアンドローゲームへようこそ。あなたは、これまで 7 回プレーしまし
た。始めますか?
User > 
```

⑦. Alexaスキル開発(CLI編)

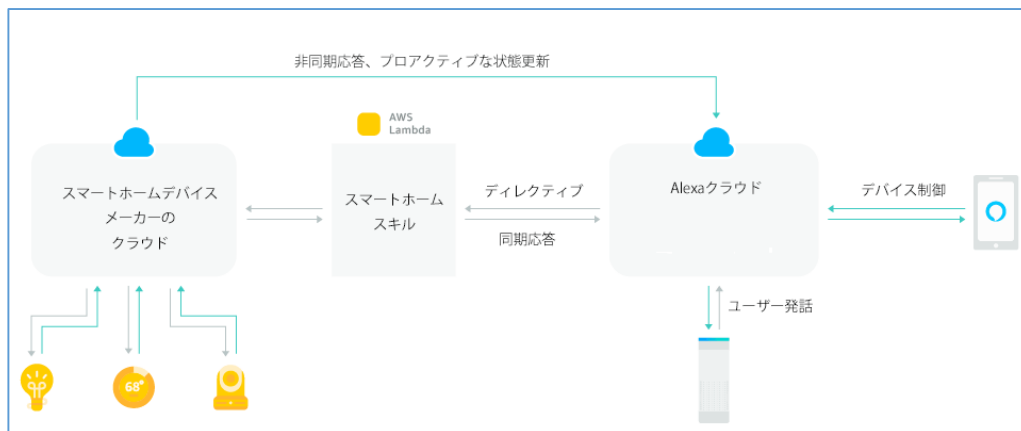


Alexa skill

Piのフォルダ位置
/home/pi/alexa/smarthome

D) SmartHome pythonで作成

- プリビルドモデルで一番利用されているスマートホームのスキル、ドキュメントは以下のURL
- <https://developer.amazon.com/ja/docs/smarthome/understand-the-smart-home-skill-api.html>
- 使い方
 1. [指定のフォルダで, requestsにより関連のプログラムなどをインストール](#)
 2. [lambda関数にアップするZipファイルを作成し、アップロード](#)
 3. [アカウントリンク用LWA\(login with amazon\)作成](#)
 4. [デバイス、シーンなどの試験の実施](#)



⑦. Alexaスキル開発(CLI編)



Alexa skill

D) SmartHome pythonで作成

- プリビルドモデルで一番利用されているスマートホームのスキル
2. Lambdaへアップロード
- [Piで作成した](#)、lambda.zipファイルのアップロード
 - ARNを確認します。Developer consoleにコピーします

The screenshot shows the AWS Lambda Developer Console for a function named 'smarthome_fc'. The 'zip ファイルをアップロード' (Upload zip file) option is selected in the 'コードエントリタイプ' (Code entry type) dropdown menu. The console displays the function's configuration, including the ARN, runtime (Python 3.6), and handler (lambda_function.lambda_handler). The code editor shows the lambda_handler function code.

Alexa開発

⑦. Alexaスキル開発(CLI編)



Alexa skill

D) SmartHome pythonで作成

- プリビルドモデルで一番利用されているスマートホームのスキル

4. デバイス、シーンなどの試験の実施

- テスト:コンソールでテストを行います。
- デバイスがつながっていないとテストできません。[弊社のスマートホーム製品](#)を使って、デバイス、シーンなどの試験ができます。[詳細は、弊社までお問い合わせください](#)。スキル>Wulianで検索可能

⑧. Alexa Voice Service開発

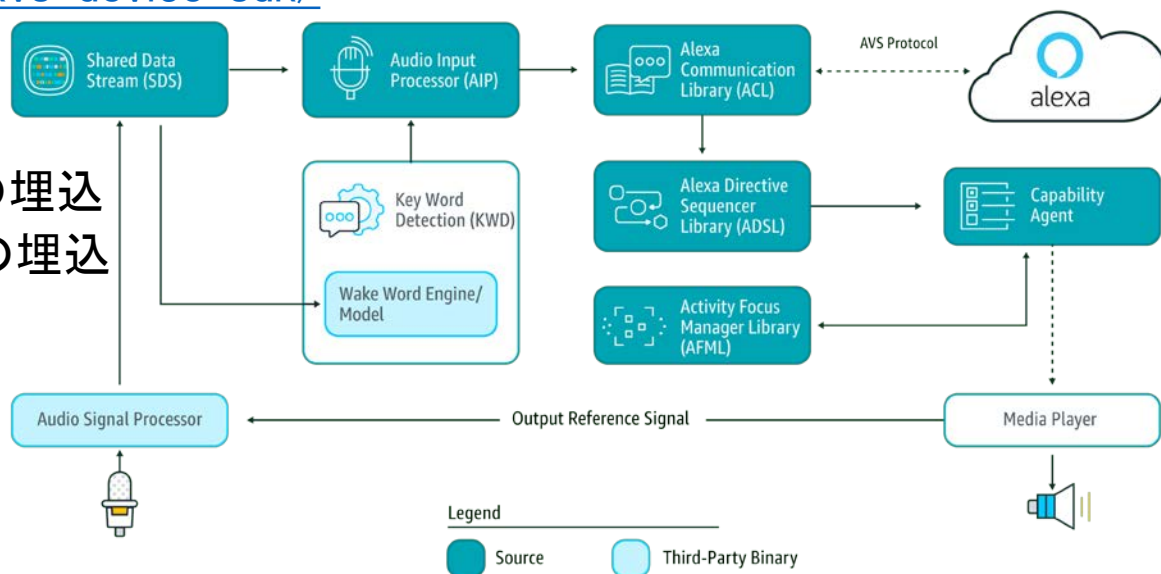
● 概要

- Alexa Voice Service(以下AVSと略します)を使って、自社製品にAlexaの機能を搭載可能になります。Echoなどのデバイスを使わないで、スピーカ、マイクを付けてAlexaを実現します。AVS Device SDKをRaspberry Piに搭載し、開発環境を提供します。また、スピーカとマイクを付属しており、Alexaの端末として利用できます。

- <https://developer.amazon.com/ja/alexa-voice-service>
- <https://alexa.github.io/avs-device-sdk/>

● 用途

- 高機能スマートスピーカ
- TVなどの家電製品への埋込
- スマートホーム製品への埋込



Alexa開発

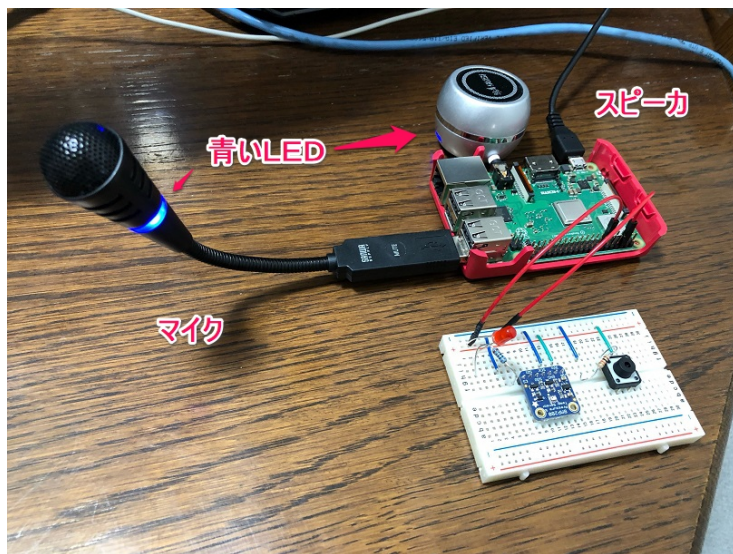
⑧. Alexa Voice Service開発

• マイク、スピーカ設定

- <https://github.com/alexavoice/avs-device-sdk/wiki/Raspberry-Pi-Quick-Start-Guide-with-Script>

• マイク、スピーカ接続

- マイク: USBにマイクを接続してください。青いLEDが点灯、赤が点灯はMuteなのでMuteを触って解除してください。
- スピーカ: イヤホンジャックに接続して、裏面のボタンを押し、青いLEDが点灯するのを確認してください。最初は、内蔵の電池をUSBミニに接続して充電してから使用してください。





⑧. Alexa Voice Service開発

- AVS sample APP動作
 - <https://github.com/alexasdk/avs-device-sdk/wiki/Raspberry-Pi-Quick-Start-Guide-with-Script>
 - コマンド画面でAuthorized, Alexa is currently idleが出れば、正常に動作中
 - マイクに向かって、Alexa、今日の天気は？と呼びかけて見ましょう。(Listeningになれば、AlexaのWakeupを認識した状態になります)

```
pi@raspberrypi: ~/alexasdk
ファイル(F) 編集(E) タブ(T) ヘルプ(H)
Press 'q' followed by Enter at any time to quit the application.
#####
# Connecting... #
#####
# Authorized! #
#####
# Alexa is currently idle! #
#####
# Client not connected! #
#####
# Alexa is currently idle! #
#####
```

本画面は、ログを省略したもので、実際のものとは違います。
\$ sudo bash startsnowboy.sh

```
pi@raspberrypi: ~/alexasdk
ファイル(F) 編集(E) タブ(T) ヘルプ(H)
#####
# Alexa is currently idle! #
#####
# Listening... #
#####
# Thinking... #
#####
#####
# RenderTemplateCard
# Focus State : FOREGROUND
# Template Type : WeatherTemplate
# Main Title : 日本寿町
#####
# Speaking... #
#####
```

Alexaと呼びかけ

今日の天気は？

Alexaが回答中

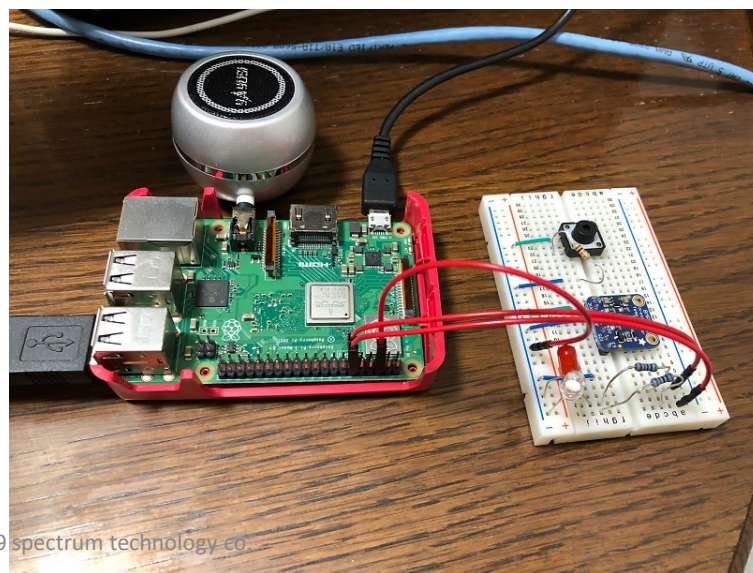


⑧. Alexa Voice Service開発

- AVS sample APP LED試験
 - <https://developer.amazon.com/ja/docs/alexa-voice-service/indicate-device-state-with-leds.html>
 - 起動時に青LED、ミュートで赤LED点灯: 接続すると点灯します
 - GPIO: 17 - 赤LED 1KΩ抵抗
 - GPIO: 18 - 青LED 1KΩ抵抗
 - GPIO: GND - GND (マイナス)
 - /home/pi/alexa/avs-device-sdk/SampleApp/src/main.cpp, Uimanger.cppなど変更済、上記ドキュメントどおり

Piのフォルダ位置
/home/pi/alexa/
\$ sudo bash startsample.sh

ブレッドボード、LEDなど必要なキットは、個別に販売していますので
[お問い合わせください](#)



Alexa開発

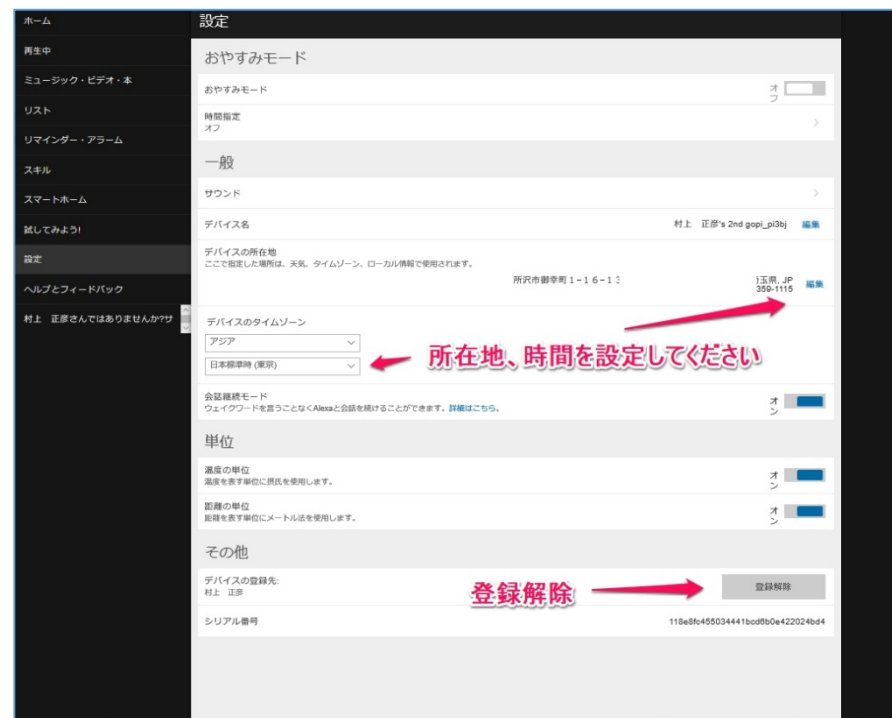
⑧. Alexa Voice Service開発



Alexa voice service

• デバイス確認

- <https://alexa.amazon.co.jp/spa/index.html#settings>
- 登録したPiのAlexa端末の確認を上記のURLから実施
- デバイスがオンラインになっていることを確認
- 端末を選択して、場所、時間の設定を行ってください。
- 登録解除も可能です。



⑧. Alexa Voice Service開発

• Wakeup Word設定(Snowboy)LED点灯

- <http://docs.kitt.ai/snowboy/#access-microphone>

- 初期設定は、起動の言葉は“Alexa”を設定していますが、カスタマイズが可能です。
- GPIOにLEDを、1KΩの抵抗を入れて接続します。

```
$ python demo_led.py alexa.umd1
```

alexaのテストとLED点灯

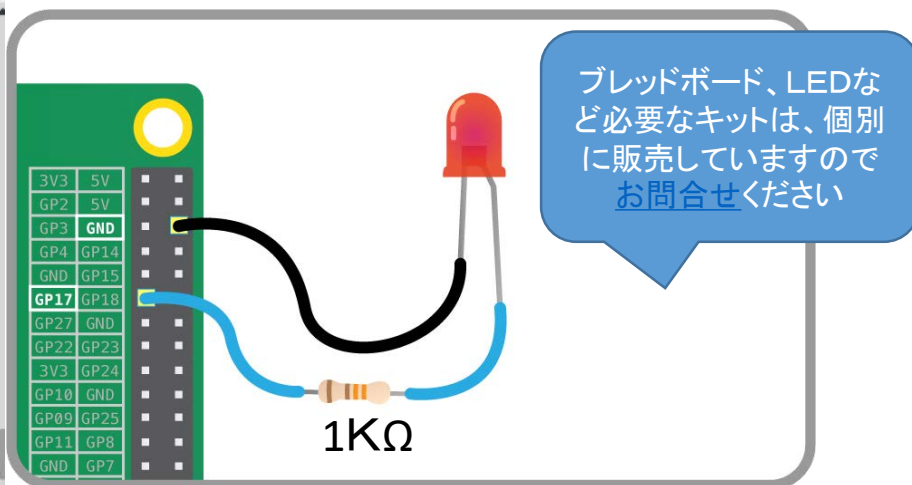
Piのフォルダ位置

`/home/pi/alexa/third-party/snowboy/examples/Python`

```
$ python demo_led.py alexa.umd1
```

```
pi@raspberrypi: ~/alexa/third-party/snowboy/examples/Python
ファイル(F) 編集(E) タブ(T) ヘルプ(H)
jack server is not running or cannot be started
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for -1, skipping unloc
k
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for -1, skipping unloc
k
Listening... Press Ctrl+C to exit
INFO:snowboy:Keyword 1 detected at time: 2019-05-14 08:38:25
^Cpi@raspberrypi:~/alexa/third-party/snowboy/examples/Python $ python demo_led.p
alexa.umd1
Cannot connect to server socket err = No such file or directory
Cannot connect to server request channel
jack server is not running or cannot be started
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for -1, skipping unloc
k
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for -1, skipping unloc
k
Listening... Press Ctrl+C to exit
/home/pi/alexa/third-party/snowboy/examples/Python/light.py:8: RuntimeWarning: T
his channel is already in use, continuing anyway. Use GPIO.setwarnings(False) t
o disable warnings.
  GPIO.setup(self.port, GPIO.OUT)
INFO:snowboy:Keyword 1 detected at time: 2019-05-14 08:42:08
INFO:snowboy:Keyword 1 detected at time: 2019-05-14 08:42:18
```

LEDランプ点灯



Alexa開発

⑧. Alexa Voice Service開発



Alexa voice
service

• Wakeup Word設定(Snowboy)カスタマイズ

- <https://snowboy.kitt.ai/dashboard#>
- 上記サイトにアクセス、ログインします。
- 3回録音し、ファイルをダウンロード。

\$ python demo.py masa.pmdl “まさ”のテスト

Piのフォルダ位置

/home/pi/alexa/third-party/snowboy/examples/Python

\$ python demo.py masa.pmdl

