



# Documentation

## Projet Snips

Révision	Date	Désignation
1	22/06/2018	Création du document



## Table des matières

1. Introduction.....	4
a. Scénario d'utilisation.....	4
b. Réalisations possibles.....	4
c. Sources de référence.....	4
d. Matériels utilisés.....	5
2. Environnement de travail.....	6
a. Travailler sous Linux.....	6
b. Travailler sous Windows 10.....	6
3. Setup.....	7
a. Interface en ligne de commande : Sam.....	7
i. Prérequis.....	7
ii. Installation.....	7
iii. Connexion à la carte.....	7
b. Raspberry Pi.....	8
i. Installation de l'OS Raspbian.....	8
ii. Installation de la plateforme Snips.....	8
c. Orange Pi Zéro +2 H5.....	9
d. Orange Pi Zéro +2 H3.....	9
i. Installation du système d'exploitation Armbian Stretch.....	9
ii. Configuration du système d'exploitation Armbian Stretch.....	9
iii. Pré-installation de Snips.....	13
iv. Installation de Snips.....	14
e. Finalisation de l'installation.....	15
4. Création d'un système de commande vocale.....	16
a. Assistant.....	16
b. Skill.....	17
i. Création d'un skill : première étape.....	17
ii. Création d'un skill : intention.....	18
iii. Création d'un skill : action.....	20
iv. Action : intent.....	21
v. Action : GPIO (Raspberry Pi).....	22



---

vi. Action : GPIO (Orange Pi).....	23
5. Utiliser un système de commande vocale.....	25
a. Connexion.....	25
b. Associer un script par Github.....	25
c. Installer un assistant.....	26
d. Installer des actions.....	26
e. Modifier le script.....	26
6. Débogage.....	27
a. Démarrage des services Snips.....	27
b. Script.....	28
7. Aller plus loin.....	29
a. Relais.....	29
b. Port USB.....	29



## 1. Introduction

Sur la base du système Open Source Snips, un système de commande vocale a été créé avec une règle d'allumage et d'extinction d'un contact sec. Cela a été fait sur une base de Raspberry pi (ou Orange Pi) avec un [Shield Respeaker Hat](#) et un module relais.

### a. Scénario d'utilisation

- Un utilisateur formule une commande à voix haute dans le micro
- Le système répond par le biais des écouteurs
  - en demandant de reformuler si la commande n'était pas claire
  - en s'excusant si la commande n'est pas réalisable (optionnel)
  - en confirmant l'exécution de la commande (bien reformuler la commande dans son intégralité)
    - l'exécution de la commande est simulée par l'actionnement du relais

### b. Réalisations possibles

- Créer une carte remplaçant le Shield Respeaker Hat avec les micro et le relais
- Ajouter un port USB dont l'alimentation est pilotée par le relais pour permettre le contrôle d'objets alimentés par USB
- Modifier le port USB en USB type C respectant la norme à 60W

### c. Sources de référence

- Snips - [Getting Started](#)
- Snips – [Documentation](#)
- Snips – [Snips Weather TTS \(Demo\)](#)
- Raspberry Pi – [Documentation Usage GPIO](#)
- Raspberry Pi – [Python et le port GPIO](#)



- Respeaker 2-Mics HAT
  - [Wiki](#)
  - [Schéma](#)
- [Tutoriel Speaker](#)
- [Orange Pi – Automatiser Règle GPIO](#)
- [Accessing GPIO](#)
- [Installer Mosquitto #1](#)
- [Installer Mosquitto #2](#)
- [Sudoer](#)
- [Activation du micro](#)
- [Dépendance](#)
- [KY-019 5V Relay Module](#)

## d. Matériels utilisés

- Raspberry Pi avec SD (OS installé ) et l'écran LCD
- Shield Respeaker
- Orange Pi Zéro +2 H3 / H5 et leur shield
- Écouteur avec micro
- Breadboard / LEDs / Composants passifs / câbles.



## 2. Environnement de travail

Pour utiliser la plateforme *Snips*, il est nécessaire de travailler en *ssh* depuis un ordinateur afin de programmer les cartes.

### a. Travailler sous Linux

Cf. **2. Environnement de Travail** – **b. Travailler sous Windows**

Cf. Snips - [Getting Started](#)

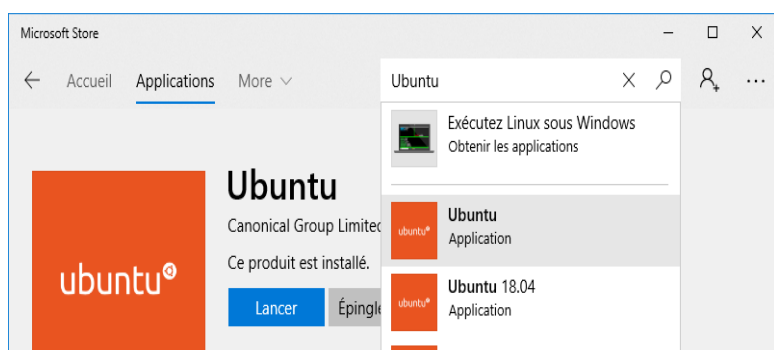
### b. Travailler sous Windows 10

La plateforme *Snips* est basée sur *Linux terminal*. Il faut donc quelques étapes préparatoires afin de simuler l'environnement Linux afin d'utiliser *Snips*.

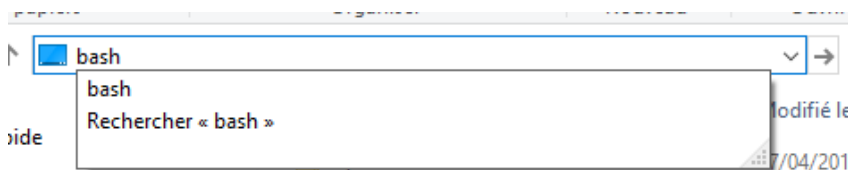
1. Ouvrir Windows *PowerShell* en tant qu'administrateur (touche Win + x, a) et taper la commande suivante :

```
> Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux
```

2. Choisir la distribution de *Linux* depuis le *Microsoft Store*



3. Taper « *bash* » dans un barre d'adresse.



4. Changer le mot de passe afin d'utiliser la commande `sudo` :

Cf. <https://docs.microsoft.com/en-us/windows/wsl/user-support>



(Source : <https://docs.microsoft.com/en-us/windows/wsl/install-win10>)

### 3. Setup

Afin d'utiliser la plateforme *Snips*, **il faut un système d'exploitation Linux Debian ou sa variation**. Attention, la distribution Ubuntu n'est pas supportée en ce moment (07/2018).

Afin de faire fonctionner *Snips*, il faut passer par l'interface en ligne de commande (CLI) *Sam*. Il est nécessaire de la configurer afin de utiliser un assistant vocal.

Pour réaliser cette tâche, il est possible de travailler sous Windows 10. Il faut d'abord utiliser le logiciel *SD Memory Card Formatter*, fourni par **SD Association**, afin de formater la carte puis *Win32DiskManager* afin d'écrire l'image du système d'exploitation dans la carte.

Sous Linux, utiliser *gparted* afin de formater la carte en format FAT32 puis :

```
dd if=/CHEMIN/NOM_DE_IMAGE.img of= /dev/sdX
```

/dev/sdX est la carte SD. Identifier la avec la commande `sudo fdisk -l`.

#### a. Interface en ligne de commande : Sam

Le CLI *Sam* commande la plateforme *Snips* depuis le PC et permet aux utilisateurs de créer, gérer et utiliser des assistants, configurer les matériels informatiques et afficher le journal d'événements de l'assistant en usage.

Cette tâche est réalisée sur le PC.

##### i. Prérequis

Logiciel *Node* dont la version est supérieure ou égale à v7.5.0 et *NPM*.

##### ii. Installation

```
$ sudo npm install -g snips-sam
```

##### iii. Connexion à la carte

Selon le tutoriel, il est possible d'exécuter la commande suivante afin de détecter une carte:

```
$ sam devices
```

Afin de pouvoir détecter la carte, il faut que la carte soit dans le même réseau que le PC (connectée au même Wifi que le PC ou par un câble Ethernet).

Cette commande n'a pas fonctionné dans le cadre de ce projet, la commande de remplacement est la suivante :

```
$ sam connect [HOST_NAME_OR_IP]
```

Afin de trouver un HOST\_NAME ou une adresse IP, utiliser la commande `hostname` ou `ifconfig` dans *Linux terminal*.



## b. Raspberry Pi

### i. Installation de l'OS Raspbian

Installer *Raspbian* d'après le tutoriel (Cf. [Snips – Getting Started](#)) et choisir [Raspbian Stretch with Desktop](#). Pour l'installation, une carte SD dont la capacité est au moins 4Go est nécessaire. Par défaut, un utilisateur **pi** est ajouté et son mot de passe est **raspberrypi**. Le fichier *sudoer* est déjà configuré dans la manière à ce que la commande **sudo** ne demande pas de mot de passe.

En revanche, il faut configurer wifi avec `sudo raspi-config` et choisir le pays où l'utilisateur habite afin de se connecter au PC.

### ii. Installation de la plateforme Snips

Ensuite, installer la plateforme *Snips* dans Raspberry Pi depuis le PC grâce à la commande suivante :

```
$ sam init
```

Enfin, il faut choisir un haut-parleur et un micro que la plateforme *Snips* pourra utiliser. Cela peut être simplement réalisé par la commande suivantes :

```
$ sam setup audio
```

Si le Shield Respeaker Hat est utilisé, dire « Yes » à l'option « Is it a Snips Maker Kit ». Ensuite, choisir *seed2micvoicec*.

Si le haut-parleur et le micro du shield ne sont pas détectés, il est nécessaire d'installer le pilote. (Cf. <https://github.com/respeaker/seeed-voicecard>) En revanche, le pilote est, en général, déjà installé dans *Raspbian*.

```
lmk@DESKTOP-UUJBKND:/mnt/c/Windows/System32$ sam setup audio
? Is it a Snips Makers Kit? Yes
[+] Installing ReSpeaker 2 mic hat
[+] Installing ReSpeaker Snips skill. This could take a while.
Rebooting device. . . done
[+] Device has rebooted
[+] Found only 1 capture device named: card 1: seeed2micvoicec [seeed-2mic-voicecard], device 0: bcm2835-i2s-wm8960-hifi w
m8960-hifi-0 []
i Using this device to capture sound
? Found 3 interfaces to output sound.
  Choose the one you want to use as your speaker card 1: seeed2micvoicec [seeed-2mic-voicecard], device 0: bcm2835-i2s-w
m8960-hifi wm8960-hifi-0 []
[+] Installed /etc/asound.conf
Setting to volume to 90% if possible
Restarting snips-audio-server service done
i You can test your microphone & speaker with sam test microphone / speaker
lmk@DESKTOP-UUJBKND:/mnt/c/Windows/System32$
```

Afin d'utiliser les pins GPIO par un script de Python, installer ou mettre à jour la bibliothèque RPi.GPIO :

```
$ sudo apt-get update
$ sudo apt-get install python-rpi.gpio python3-rpi.gpio
```





## c. Orange Pi Zéro +2 H5

Malgré avoir testé avec les distributions officielles d'*Orange Pi* et les distributions d'*Armbian* (expérimental - 07/2018), la carte **Orange Pi Zéro +2 H5** n'a pas bien fonctionné. La plateforme *Snips* ne fonctionne donc pas sur cette carte.

## d. Orange Pi Zéro +2 H3

### i. Installation du système d'exploitation Armbian Stretch

Le fonctionnement de la plateforme *Snips* est vérifié dans le système d'exploitation *Armbian Stretch* (mainline kernel 4.1.4.y). La distribution est disponible sur le [site web officiel Armbian](http://www.armbian.com). Le mot de passe **root** est par défaut **1234** et il doit être changé dès que l'utilisateur se connecte.

Pour l'image disponible dans notre blog (<https://letmeknow.fr/blog/>), le mot de passe **root** est **letmeknow**. Un utilisateur **pi** est ajouté et son mot de passe est **1234**.

### ii. Configuration du système d'exploitation Armbian Stretch

Pour pouvoir installer et utiliser la plateforme *Snips*, les fonctionnalités nécessaires doivent être activées ou configurées. Elles sont les suivantes :

- Sudoer
  - Dans cette distribution, les utilisateurs non-root ne peuvent pas exécuter les programmes par défaut avant de changer la configuration. Utiliser la commande :

```
$ sudo visudo
```

si cela ne marche pas, se connecter à l'utilisateur **root** en utilisant la commande **su** et exécuter **visudo**. Le fichier **sudoer.tmp** est ouvert dans un éditeur de texte *nano*. Ajouter la ligne ci-dessous à la fin :

```
$USER ALL=(ALL) NOPASSWD:ALL
```

Quitter en tapant **Ctrl + X** et puis **Y + Entrée** afin de enregistrer.

- Wi-Fi
  - Contrairement à la distribution Raspbian, il n'est pas nécessaire de préciser le pays dans quel l'utilisateur l'est.

```
$ sudo armbian-config
```

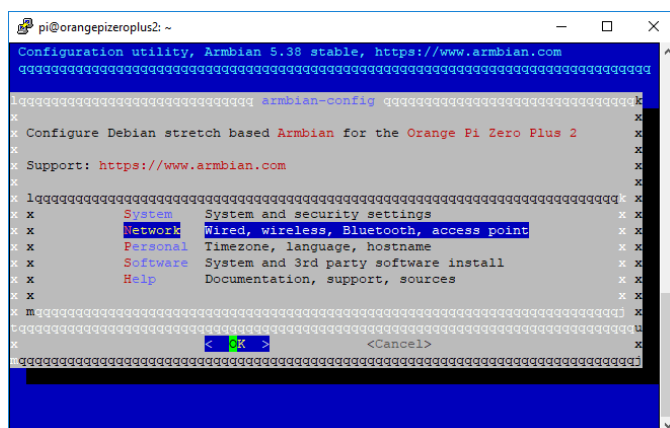
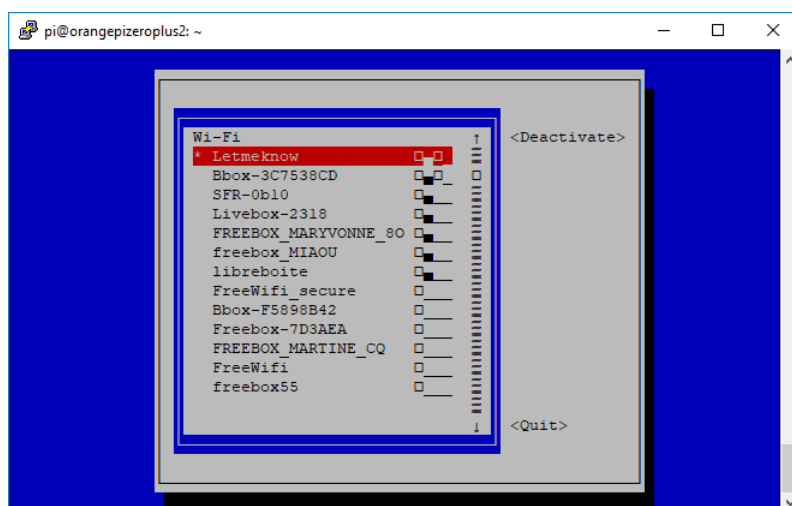
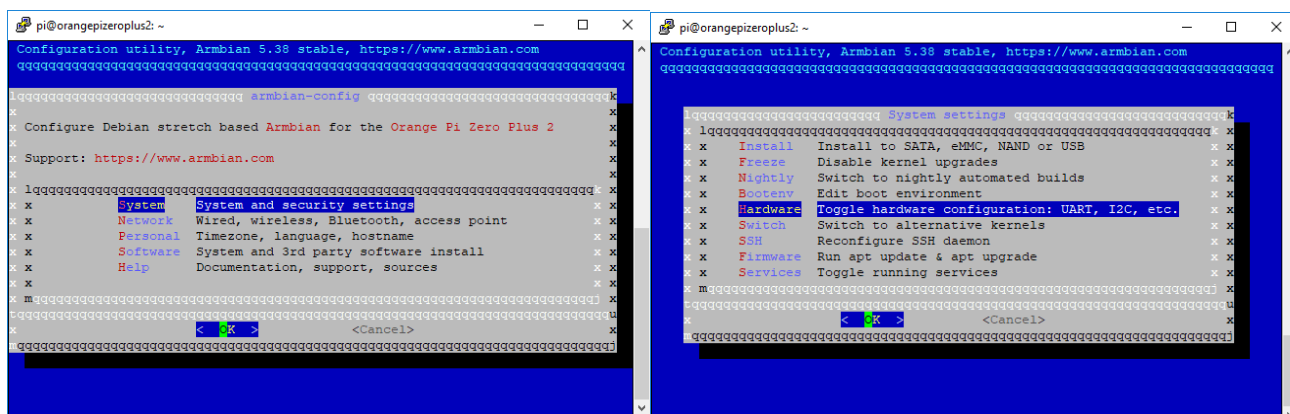


Illustration 1: armbian-config

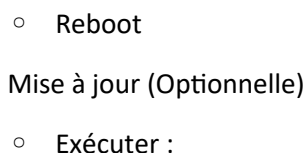
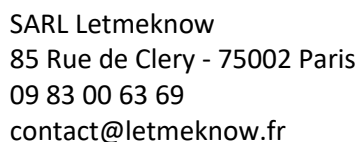
- Sélectionner « Network » et puis « Wi-Fi ».



- Sélectionner votre propre SSID et entrer le mot de passe. Enfin, il faut **activer** le Wi-Fi et quitter.
- Analog-codec (pour l'audio)
  - Continuer sans quitter ou reutiliser la commande `sudo raspi-config` et sélectionner « System » et puis « Hardware »



- Activer « analog-code » en tapant la touche **Espace**. « Save » et quitter.

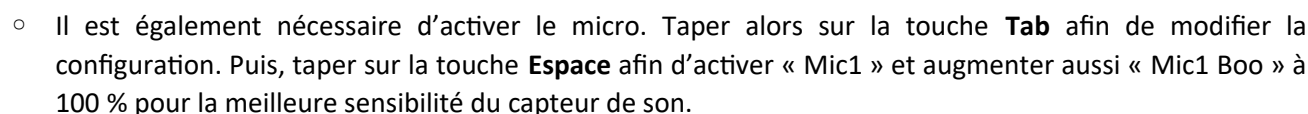


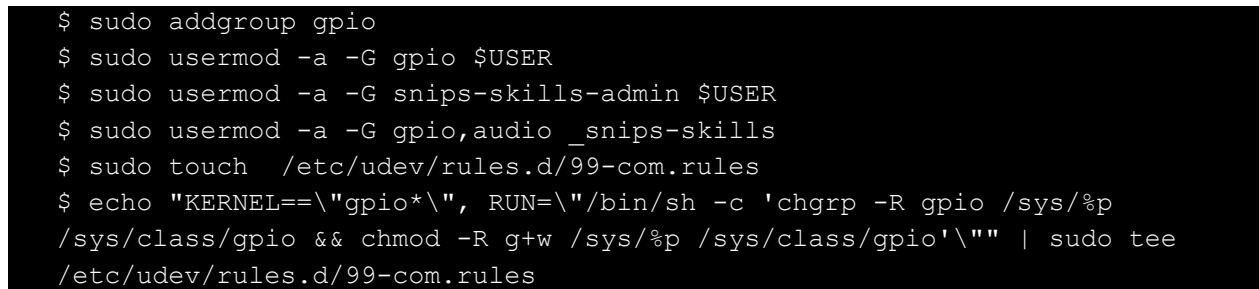
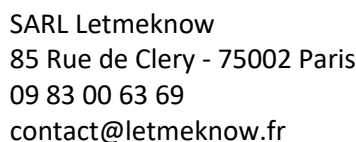
## Alsamixer

- Exécuter :

```
$ alsamixer
```

- Afin de bien configurer le haut parleur, augmenter « Line Out » (volume) à 100 % et sélectionner soit Mono Différentiel soit Stéréo en utilisant **touche flèche** selon la préférence.





- GPIO
    - La plateforme *Snips* doit d'abord être installée.
    - Utiliser **gpio\_config.sh** pour la configuration rapide.
    - La plateforme Snips n'a pas de droit afin de commander les pins GPIO. Afin de commander, il faut pouvoir écrire sur les fichiers dans `/sys/class/gpio/` mais ce privilège n'appartient à que son propriétaire. Alors, il est nécessaire de créer un groupe **gpio** et ajouter les fichiers à ce groupe. Ensuite, autoriser l'accès à écrire les fichier à ceux qui appartient à ce groupe. Automatiser cette tâche pour chaque pin GPIO en utilisant la règle suivante :
- ```
$ sudo addgroup gpio
$ sudo usermod -a -G gpio $USER
$ sudo usermod -a -G snips-skills-admin $USER
$ sudo usermod -a -G gpio,audio _snips-skills
$ sudo touch /etc/udev/rules.d/99-com.rules
$ echo "KERNEL==\"gpio*\", RUN=\"/bin/sh -c 'chgrp -R gpio /sys/%p
/sys/class/gpio && chmod -R g+w /sys/%p /sys/class/gpio'\"" | sudo tee
/etc/udev/rules.d/99-com.rules
```
- Reboot



### iii. Pré-installation de Snips

Contrairement à Raspberry Pi, Il faut installer *Mosquitto* pour Orange Pi car *Snips* ne démarre pas sans *Mosquitto*.

Il faut d'abord installer les dépendances nécessaires et puis mettre à jour le paquet afin de pouvoir installer le logiciel en utilisant `apt-get install`.

Les dépendances nécessaires :

- libssl 1.0.0 (**Prérequis pour libwebsocket3**)
- libwebsockets3 1.2.2.1

Utiliser `moquitto_install.sh` pour une installation rapide.

Ou utiliser les commandes afin de les installer :

```
$ wget
http://ftp.fr.debian.org/debian/pool/main/libw/libwebsockets/libwebsockets3_1.2.2-
1_armhf.deb

$ wget http://ftp.fr.debian.org/debian/pool/main/o/openssl/libssl1.0.0_1.0.1t-
1+deb8u8_armhf.deb

$ sudo dpkg -i ~/mosquitto/libssl1.0.0_1.0.1t-1+deb8u8_armhf.deb
$ sudo dpkg -i ~/mosquitto/libwebsockets3_1.2.2-1_armhf.deb
```

Afin d'installer Mosquitto :

```
$ wget http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key
$ sudo apt-key add mosquitto-repo.gpg.key
$ cd /etc/apt/sources.list.d/
$ sudo wget http://repo.mosquitto.org/debian/mosquitto-jessie.list
$ sudo apt-get update
$ sudo apt-get install mosquitto
$ sudo apt-get install mosquitto-clients
```

**Remarque :**

- **Les dépendances nécessaires pour Mosquitto peuvent être changées selon la mise à jour future (07/2018).**
- **Vérifier le démarrage du service Mosquitto en utilisant la commande `sudo systemctl | grep MQTT`. Sinon, ouvrir le fichier `/etc/init.d/mosquitto` et changer la configuration :**
  - `--exec ${DAEMON} -- -c /etc/mosquitto/mosquitto.conf → --exec ${DAEMON} -- -d`



#### iv. Installation de Snips

Installer d'abord la plateforme *Snips* dans Orange Pi depuis le PC grâce à la commande suivante :

```
$ sam init
```

Contrairement à Raspberry Pi, cela n'installe pas complètement le logiciel.

L'installation automatique ne fonctionne pas dans cette carte. Donc, l'installation manuelle des services suivants est exigée. Cela peut être réalisé par `sudo apt-get install NOM DE SERVICE` ou `snips_install.sh`

- Snips service à installer
  - **snips-analytics**
  - **snips-hotword**
  - **snips-nlu**
  - snips-queries
  - **snips-watch**
  - **snips-asr**
  - **snips-dialogue**
  - snips-audio-client
  - **snips-kaldi-atlas**
  - **snips-audio-server**
  - snips-makers-tts
  - **snips-tts**

#### Remarque :

- *Pour Snips, les services en gras sont essentiels mais ceux qui ne le sont pas peuvent être optionnels (à vérifier).*

Enfin, il faut choisir un haut-parleur et un micro que la plateforme *Snips* pourra utiliser. Cela peut être simplement réalisé par la commande suivante :

```
$ sam setup audio
```

Orange Pi n'utilise pas de « Snips Maker Kits », choisir donc « No » pour l'option « Is it a Snips Makers Kit ». Choisir H3 Audio Codec pour la carte son.

```
? Is it a Snips Makers Kit? No
Found only 1 capture device named: card 0: Codec [H3 Audio Codec], device 0: CDC PCM Codec-0 []
i Using this device to capture sound
? Found 2 interfaces to output sound.
  Choose the one you want to use as your speaker card 0: Codec [H3 Audio Codec], device 0: CDC PCM Codec-0 []
Installed /etc/asound.conf
Setting to volume to 90% if possible
Restarting snips-audio-server service done
i You can test your microphone & speaker with sam test microphone / speaker
```



## e. Finalisation de l'installation

Afin d'assurer le fonctionnement correct des dispositifs, effectuer les commandes suivantes :

```
$ sam test speaker  
$ sam test microphone
```

Ces commandes permettent de tester respectivement le haut-parleur et le micro choisis.

Afin de voir comment l'assistant vocal fonctionne sur la plateforme *Snips*, installer l'assistant demo fourni par *Snips*.

Cf. <https://snips.gitbook.io/getting-started/install-a-demo-assistant>

Cf. <https://github.com/snipsco/snips-skill-weather-tts>



## 4. Création d'un système de commande vocale

Afin de créer un système de commande vocale, il faut créer un « assistant ». La création et gestion de l'assistant sont faites depuis le site web, <https://console.snips.ai/home>, après l'inscription.

C'est depuis cette interface qu'il est possible de configurer les actions correspondant aux intentions émises par l'utilisateur.

snips

Snips for your home  
Connect to Snips platform

EMAIL  
jon.snow@nightwatch.com

PASSWORD  
6+ characters  
[Forgot Your Password?](#)

Login

Don't have an account? [Sign Up](#)

### a. Assistant

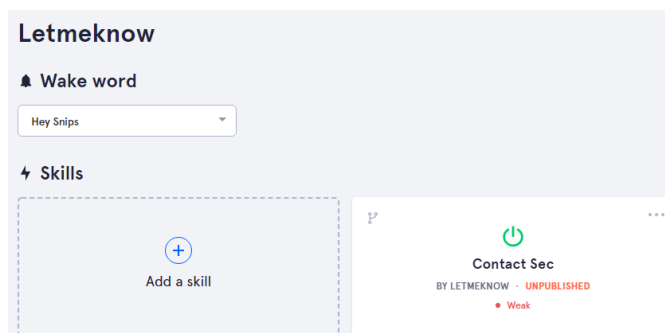
L'assistant est une interface entre l'utilisateur et les « skills » écrits par les utilisateurs. Un assistant peut gérer et choisir un skill parmi plusieurs qui correspondent à l'intention émise par l'utilisateur lors de son exécution.

Il est possible de créer un assistant depuis [le site web](#) et choisir une langue parmi 6 proposées (allemand, anglais, espagnol, français, japonais et coréen).





Après la création, ajouter un ou plusieurs skills dans l'assistant.



## b. Skill

Un skill, ou compétence, se compose d'une intention et des actions.

En résumé une compétence est la capacité de comprendre une intention et d'y répondre par une ou plusieurs actions correspondantes.

- Une intention peut représenter des commandes vocales d'un même objectif.
- Une action est ce qu'un assistant vocal doit faire lorsqu'il est commandé. L'action peut être sous la forme d'un script ou de code snippets.

### i. Création d'un skill : première étape

Cela se fait depuis le site web : cliquer sur "Add a skill" dans le gestionnaire d'assistant (Cf. [4. Création d'un système de commande vocale – a. Assistant](#)). Il y a 2 choix : soit importer un skill publié par des autres utilisateurs de *Snips* soit faire son propre skill.

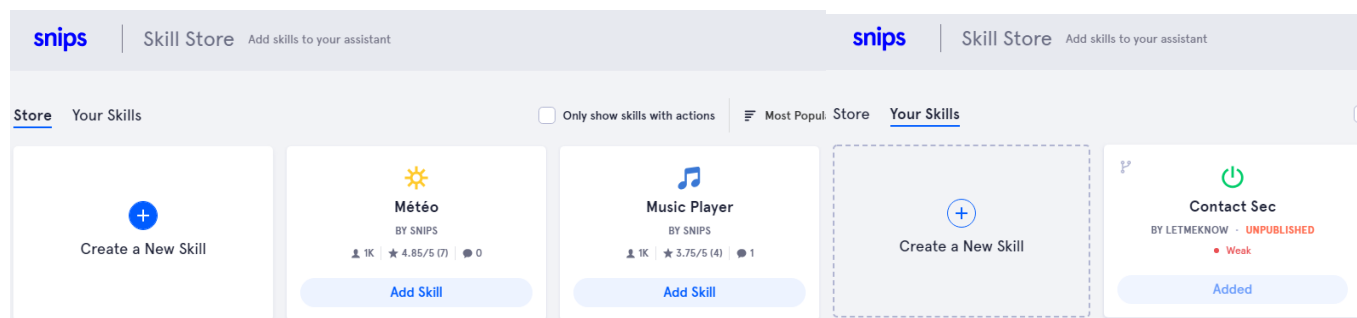



Illustration 2: Skills publiés

Illustration 3: Skill en possession



Create Skill ✕

PICTURE  
 [Choose From Library](#)


SKILL NAME Required

DESCRIPTION

[Cancel](#) [Create](#)

## ii. Création d'un skill : intention

Afin de créer une nouvelle intention, cliquer sur « + Create New Intent » après choisir un skill à modifier.

**Contact sec**  
Letmeknow · weak quality  
Une règle d'allumage et d'extinction d'un contact sec.

2 Intents

[+ Create New Intent](#) [Import Intents](#)

- Allumage** · Letmeknow:Allumage  
4 training examples · 1 slot
- Extinction** · Letmeknow:Extinction  
2 training examples · 1 slot

Home > Letmeknow > Contact sec > Allumage

**Allumage** ✕

INTENT NAME

INTENT DESCRIPTION (500 characters max)

1 Slot ⊕

| NAME  | TYPE                    | SLOT REQUIRED?                          |
|-------|-------------------------|-----------------------------------------|
| Verbe | verbe/Verbeult<br>Verbe | <input type="checkbox"/> Slot Required? |

4 Training Examples ⊕

Automatically generate additional training examples [Generate](#)

Type your training example... [Add Training Example](#)

- Allumage** la lumière
- Allumage** la lumière
- Mettre** la lumière

Une intention est composée de 3 éléments

- nom / description
- « slot »
- Exemples d'entraînement.

Tout d'abord, il faut définir le nom d'une intention. Cela est utilisé lors du codage d'un script. La description est optionnelle.

Home > Letmeknow > Contact sec > Allumage

**Allumage** ●

INTENT NAME

INTENT DESCRIPTION (500 characters max)

Ensuite, créer des slots. Un slot représente des mot-clés à reconnaître dans la commande vocale. Par exemple, les verbes « allumer », « mettre » et « éclairer » peuvent être représentées par un slot qui s'appelle « Verbe ». Les noms



« chambre », « cuisine » et « garage » peuvent être représentés par un slot qui s'appelle « Salles ». Ainsi, il n'est plus nécessaire de faire des exemples d'entraînement compréhensifs. Ce sujet sera être traité plus tard.

Texte 1: Cliquer sur « + Add Slot » afin de créer un slot

- Nom : cela peut être aléatoire.
- Type : la catégorie des mots-clés
  - Il y a des types de slot prédéfini :
    - S'il est nécessaire de reconnaître les nombre dans une commande vocale, choisir « snips/number »
    - Idem pour température, pourcentage, etc.
  - Si un nouveau type de slot est nécessaire, choisir « snips/default » ou cliquer sur « New Slot Type ».
    - New Slot Type :

- « Generate Values » : Cela permet de trouver une catégorie désirée (une liste des mots-clés) à partir des exemples de la catégorie.
- Dans « Manual Slot Type Values », ajouter manuellement les mots qui appartiennent à la catégorie mais qui ne sont pas dans la liste des mots-clés.
  - snips/default : Cela représente aucune catégorie.
- « Slot required ? » : Une question à poser si le slot n'est pas détecté dans une commande vocale.



Enfin, il faut saisir les exemples d'entraînement. Autrement, *Snips* ne peut pas comprendre la commande vocale.

Afin de faire un exemple, saisir une phrase dans le boîte de texte où « Type your training example... » est marqué et puis cliquer sur « + Add Example ».

**Remarque : La génération automatique est un service payant.**

Tout d'abord, sélectionner un mot afin de le relier avec un slot.

| NAME       | TYPE                    | SLOT REQUIRED?                              |
|------------|-------------------------|---------------------------------------------|
| secondTerm | snips/number<br>builtin | <input checked="" type="checkbox"/> Quel es |
| firstTerm  | snips/number<br>builtin | <input checked="" type="checkbox"/> Quel es |

Les mots mis en évidence sont associés avec les slots de la même couleur.

Le type de slot permet de reconnaître les nombres sans saisir les exemples exhaustifs comme :

*fais le calcul de 1 plus 1, fais le calcul de 1 plus 2, ..., fais le calcul de 1 plus 99999*

### iii. Création d'un skill : action

Dans le cadre de ce projet, un script a été utilisé afin de définir l'action.

( 03/07/2018 - La documentation officielle mise à jour de home assistant :

<https://snips.gitbook.io/documentation/console/actions/home-assistant> )

Action Type

Need help? [Skill Action docs](#)

- ☐ None  
No action associated to intents
- ☒ Github  
Get skill actions from a Github repository
- ☐ Code Snippets  
Write action code for each intent
- ☐ Home Assistant Component  
Bind intents to a pre-made component

Il y a également deux manières d'écrire un script. La première méthode est « code snippets ». La méthode « code snippets » permet d'écrire un script indépendant pour chaque intention. Cela est simple et rapide mais il manque de



flexibilité au niveau de gestion des erreurs ou gestion des cas imprévus. (un exemple d'un code : <https://snips.gitbook.io/documentation/console/actions/snippets>)

La méthode utilisant Github permet de gérer toutes les intentions en un seul script. Cela permet de gérer les exceptions d'une manière plus flexible. (Cf. action-contact.sec.py)

#### iv. Action : intent

Les commandes vocales sont traitées dans la fonction :

```
def intent_received(hermes, intent_message):
```

La variable **intent\_message** contient l'information d'une commande vocale analysée et enregistrée en format JSON objet :

**intent\_message**

↳ **intent** : l'Intention

↳ **intent\_name (String)** : l'identificateur de « intent » (commande vocale)



Illustration 4: L'identificateur de «intent»  
est entouré en bleu

↳ **probability (float)**: La correspondance en pourcentage entre l'intention présumée et l'intention émise par l'utilisateur.

Dans le cadre de ce projet, si le seuil de **probabilité** est inférieur à 90 %, *Snips* considère qu'il n'a pas compris et émet un message : « Je n'ai pas compris. »

Afin de faire émettre un **message** souhaité, écrire:

```
hermes.publish_end_session(intent_message.session_id, message)
```



#### v.Action : GPIO (Raspberry Pi)

Afin de commander les pins GPIOs charger la bibliothèque **RPi.GPIO** :

```
import RPi.GPIO as GPIO
```

Avant de pouvoir commander les pins GPIO, Configurer d'abord :

```
GPIO.setwarnings(False) #Pour la suppression des avertissements inutiles par l'interpréteur de Python.
```

```
GPIO.setmode(GPIO.BCM) # Commander un pin GPIO par sa propre numérotation (GPIO 12 = 12)
```

```
GPIO.setup(12, GPIO.OUT) # GPIO12 est une sortie
```

ou

```
GPIO.setmode(GPIO.BOARD) # Commander un pin GPIO par le nombre de pin (GPIO 12 = 32)
```

```
GPIO.setup(32, GPIO.OUT) # GPIO12 est une sortie
```

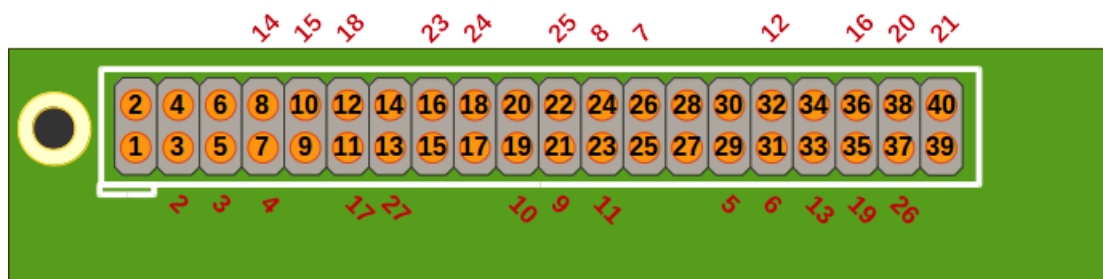


Illustration 5: En noir la numérotation GPIO.BOARD. En rouge, la numérotation GPIO.BCM  
(Source: <https://deussyss.developpez.com/tutoriels/RaspberryPi/PythonEtLeGpio/>)

Enfin, afin de commander un pin GPIO :

```
GPIO.output(12, GPIO.LOW)
```

ou

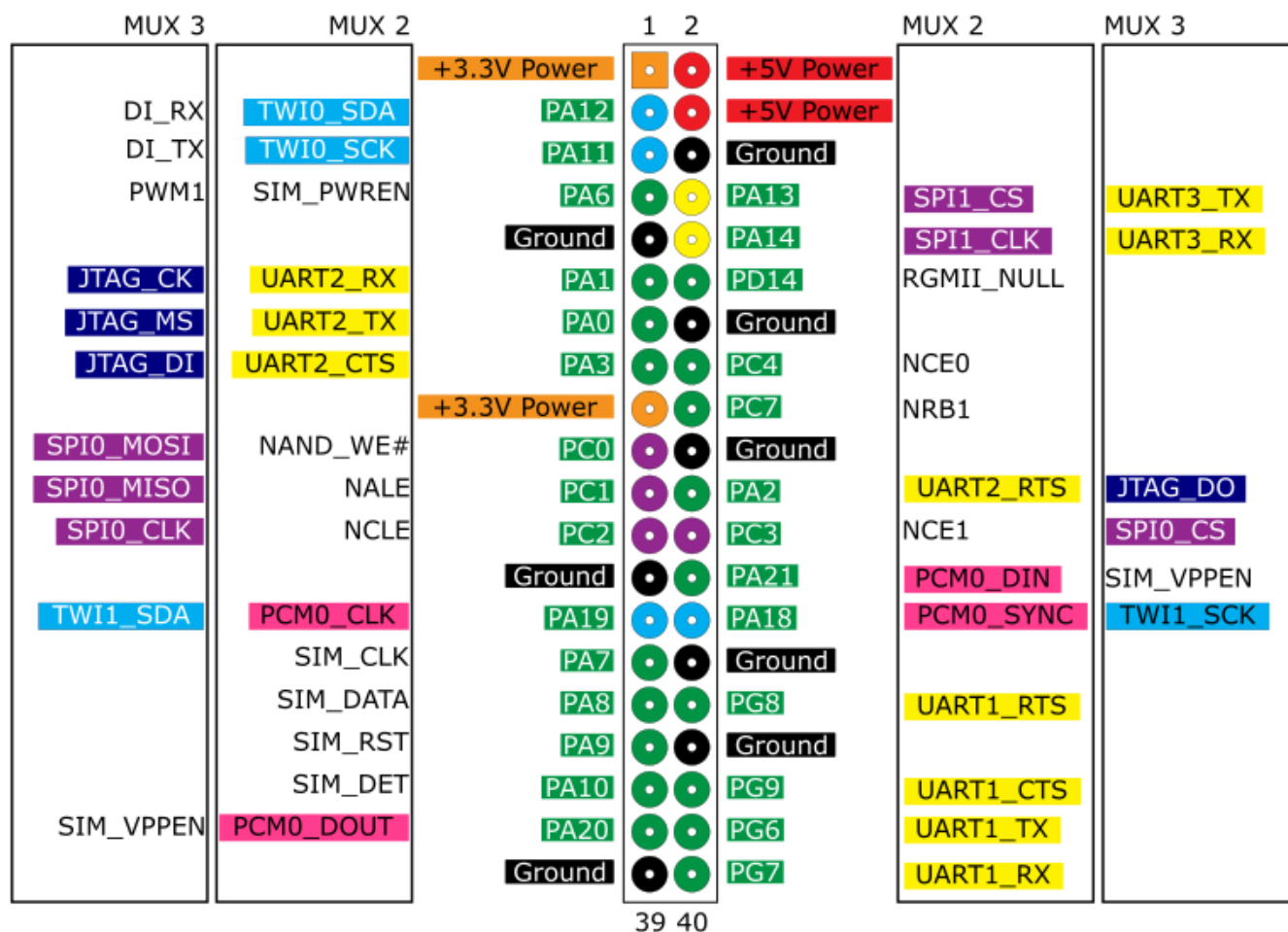
```
GPIO.output(12, GPIO.HIGH)
```



vi. Action : GPIO (Orange Pi)

Dans le cadre de ce projet, l'assistant vocal communique directement avec un GPIO au lieu d'utiliser une bibliothèque. Avant d'effectuer cette démarche, la configuration de GPIO est prérequis (Cf. [3.Setup](#) – [d.Orange Pi Zéro +2 H3](#) – [ii.Configuration du système d'exploitation Armbian Stretch - GPIO](#))

## Orange Pi (H3 SoC) GPIO - pinout



NOTE: GPIO voltage levels are 3.3V.

■ JTAG ■ I2C ■ SPI ■ +5V ■ GPIO ■ UART ■ +3.3V ■ Ground ■ I2S/PCM

Illustration 6: Source : <https://stackoverflow.com/questions/46463724/accessing-gpio-on-orangepi-pc-plus-h3-on-armbian-3-4-113-and-newer>

1. Afin d'accéder un pin GPIO, il est nécessaire tout d'abord d' « exporter » :

```
$ echo GPIO_PIN_NUM | sudo tee /sys/class/gpio/export
```

ou

```
$ echo GPIO_PIN_NUM > /sys/class/gpio/export
```



**Remarque :** La commande `sudo` ne peut pas être utilisée avec la redirection (`'>'` ou `'>>'`). Utiliser alors `sudo tee` avec un pipeline.

- `GPIO_PIN_NUM`: (la position d'une lettre - 1) \* 32 + le numéro de pin.
  - Par exemple, le `GPIO_PIN_NUM` de **PA12** est calculé dans la manière suivante :

**A = 1<sup>ère</sup> position**

**Le numéro de pin = 12**

Alors, `GPIO_PIN_NUM` = (1 - 1) \* 32 + 12 = 12

2. Définir si le pin GPIO est une sortie ou une entrée :

Configurer en tant que sortie :

```
$ echo out | sudo tee /sys/class/gpio/gpioGPIO_PIN_NUM/direction
```

Par exemple, pour le pin GPIO PA12 :

```
$ echo out | sudo tee /sys/class/gpio/gpio12/direction
```

3. Attribuer la valeur du pin GPIO :

GPIO On :

```
$ echo 1 | sudo tee /sys/class/gpio/gpioGPIO_PIN_NUM/value
```

GPIO Off :

```
$ echo 0 | sudo tee /sys/class/gpio/gpioGPIO_PIN_NUM/value
```

4. Enfin, « unexporter » le pin GPIO utilisé. (optionnel)

Afin de réaliser ces commandes dans un script écrit en python, utiliser les bibliothèques **os** et **time**.

Premièrement vérifier l'écriture aux fichiers GPIO est autorisé par :

**if os.access(gpio\_path + "/export", os.W\_OK) :**

Si la fonction **os.access** retourne **True**, écrire :

```
gpio_pin = os.open(gpio_path + "/export", os.O_WRONLY) #WRONLY : WRITE ONLY
```

```
os.write(gpio_pin, str(GPIO_PIN_NUM))
```

```
os.close(gpio_pin)
```

Écrire le code dans la même manière pour:

- `/sys/class/gpio/gpioGPIO_PIN_NUM/direction`
- `/sys/class/gpio/gpioGPIO_PIN_NUM/value`
- `/sys/class/gpio/unexport`





#### Remarque :

Ajouter un petit retard entre l'écriture à `/sys/class/gpio/export/` et `/sys/class/gpio/gpioGPIO_PIN_NUM/` direction `time.sleep(0.05)`

Il faut le temps à Orange Pi afin de créer un répertoire `/sys/class/gpio/gpioGPIO_PIN_NUM/` et les fichiers dedans.

## 5. Utiliser un système de commande vocale

### a. Connexion

Depuis le PC, relancer `bash` et saisir :

```
> sam login
? Enter email used on the console: votre.email@domain.fr
? Enter password used on the console: [input is hidden]
```

Saisir l'email et le mot de passe utilisés afin de s'inscrire à <https://console.snips.ai>

### b. Associer un script par Github

Créer un répertoire Github ([https://www.github.com/NOM\\_UTILISATEUR/NOM\\_REPERTOIRE](https://www.github.com/NOM_UTILISATEUR/NOM_REPERTOIRE))

Ensuite, il faut 3 fichiers :

- `action-*.py` : script exécuté en boucle
- `requirements.txt` : signaler d'installer les bibliothèques nécessaires.
  - Par exemple, pour un script dans Raspberry Pi :  
`hermes_python>=0.1`  
`RPi.Gpio`
- `setup.sh` : faire pouvoir exécuter un script dans un environnement virtuel de Python et installer les dépendances.

Associer ce répertoire avec un skill :



Action Type

Need help? [Skill](#) [Action docs](#)

☐ None  
No action associated to intents

☒ Github  
Get skill actions from a Github repository

☐ Code Snippets  
Write action code for each intent

☐ Home Assistant Component  
Bind intents to a pre-made component

GITHUB REPOSITORY

[Link](#)

**Remarque : Si le script ne lance pas, il faut vérifier l'existence du répertoire `/var/lib/snips/skills/NOM_REPERTOIRE_GIT`.**

## c. Installer un assistant

Lancer depuis le PC :

```
> sam install assistant
```

S'il y a plusieurs assistant, il ne faut en choisir qu'un. Les skills appartenant à l'assistant téléchargé sont également téléchargés. Si l'interface en ligne de commande *Sam* prévient qu'il faut effectuer `chmod +x`, ajouter le privilège aux scripts :

```
$ chmod +x /var/lib/snips/skills/NOM_DE_SKILL/action-*
```

## d. Installer des actions

Depuis le PC :

```
> sam install actions
```

Cette commande est principalement utile afin de mettre à jour les code snippets.

**Remarque : Si le github est associé avec le skill à mettre à jour, la mise à jour d'un skill du même nom exige effacer le répertoire du skill dans `/var/lib/snips/skills/NOM_DE_SKILL`**

## e. Modifier le script

Modifier `/var/lib/snips/skills/NOM_DE_SKILL/action-*`

Puis, lancer :

```
$ sudo service snips-skill-server restart
```

Maintenant, *Snips* exécute le script modifié



## 6. Débogage

Si le système d'allumage ne fonctionne pas, il est possible d'effectuer quelques manipulation afin de trouver le problème.

### a. Démarrage des services Snips

Lancer la commande depuis le PC :

```
> sam status
```

```
lmk@DESKTOP-UUJBKND:/mnt/c/Windows/System32$ sam status

Connected to device 192.168.1.145

OS version ..... Debian GNU/Linux 9 (stretch)
Installed assistant ..... Letmeknow
Language ..... fr
Hotword ..... hey_snips
ASR engine ..... snips
Status ..... Live

Service status:

snips-analytics ..... 0.56.4 (running)
snips-asr ..... 0.56.4 (running)
snips-audio-server ..... 0.56.4 (running)
snips-dialogue ..... 0.56.4 (running)
snips-hotword ..... 0.56.4 (running)
snips-nlu ..... 0.56.4 (running)
snips-skill-server ..... 0.56.4 (running)
snips-tts ..... 0.56.4 (running)
```

- OS version doit être **Debian** ou sa variation
- Confirmer si un assistant est aussi bien installé.
  - Sinon, Cf. [5.Utiliser un système de commande vocale – c.Installer un assistant](#)
- Si tous les services fonctionnent sans problème, le service est « **live** » tous les « service status » sont indiqués « **(running)** ».
  - Sinon,
    - exécuter `sudo systemctl | grep MQTT` et `sudo systemctl | grep snips` (depuis la carte)
    - Si l'un des deux ou les deux ne fonctionnent pas, lancer `sudo journalctl`.
    - Cf. [3.Setup – d.Orange Pi Zéro Plus 2 H3 – iii.Pré-installation de Snips](#)
- Lire un log plus détaillé : `journalctl -u SERVICE`



- Par exemple : `journalctl -u snips-skills-server`

## b.Script

Regarder soit « **code snippets** » soit `/var/lib/snips/skills/NOM_DE_SKILL/action-*.py`. Vérifier qu'il n'y a pas de faute.

Dans le cas où un script (action-\*.py) est utilisé, exécuter-le avec Python :

```
$ python /var/lib/snips/skills/NOM_DE_SKILL/action-*.py
```

Il faut afficher un message qui se rassemble à le suivant :

```
pi@orangepizeroplus2:~$ python /var/lib/snips/skills/Snips-ContactSec-OPi/action-contact.sec.py
Traceback (most recent call last):
  File "/var/lib/snips/skills/Snips-ContactSec-OPi/action-contact.sec.py", line 3, in <module>
    from hermes_python.hermes import Hermes
ImportError: No module named hermes_python.hermes
```



## 7. Aller plus loin

Il est possible de réaliser une commande vocale avec un relais ou un port USB.

### a. Relais



Illustration 7:

常开 : NO

公共端 : COM

常闭 : NC

Le relais est un interrupteur électromagnétique. Le relais est alimenté par la borne **VCC** et lorsque un signal électrique entre par la borne **IN**, un électro-aimant à l'intérieur est alimenté afin de fermer le circuit.

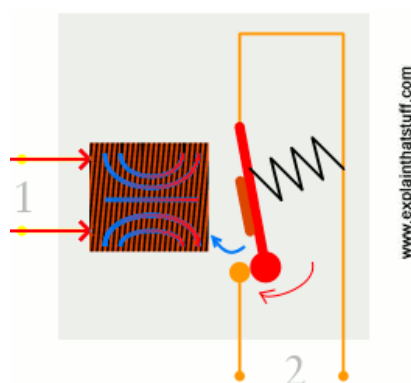


Illustration 8: Une illustration du fonctionnement d'un relais

Source: <https://www.explainthattuff.com/howrelayswork.html>

Après la fermeture de l'interrupteur, la tension de sortie à la borne NO (Normally Open) devient la tension alimentée par la borne COM. Lorsque le circuit est ouvert, la tension de sortie à la borne NC (Normally Closed) devient la tension alimentée par la borne COM.

### b. Port USB

L'alimentation du port USB ne peut pas être coupée dans Orange Pi Zéro Plus 2. Il faut un module externe d'un port USB.