PocketPlay Documentation

How to Install Snips on Raspberry Pi 3

1. Follow instructions for installation at this link but stop at step 3 “Installing a demo assistant” (use ethernet connection preferably): <https://snips.gitbook.io/documentation/installing-snips/on-a-raspberry-pi>
2. Snipsmanager is also necessary to install. Follow the directions for the python package install at this link: <https://github.com/snipsco/snipsmanager>

Caution: The snipsmanager platform is deprecated and while it should still work an updated snips management platform is currently available and should be looked into.

How to Install Custom Hotword

1. Follow instructions for installation at this link: <https://snips.gitbook.io/documentation/advanced-configuration/wakeword/personal-wakeword>
2. Make sure when you download the assistant from the snips console that the wake word is set to “personal wake word”

How to Install Custom Assistant

1. Go to [www.console.snips.ai](http://www.console.snips.ai) to download or create your assistant
2. Once the assistant is downloaded, navigate to the Downloads folder in the terminal and run this command: sudo snips-install-assistant <name of assistant>
3. The following directory will have been made or updated /opt/snips/config/assistant. The contents of this directory must be copied over into this directory /usr/share/snips/assistant. If this directory doesn’t already exist create it yourself.
4. The Snipsfile must be placed in /usr/share/snips/assistant
5. Run this command: sudo snipsmanager install
6. With everything else set up it is time to test snips by running sudo snipsmanager run

The Snips Console

The snips console at <https://snips.console.ai> is where the assistant is made and its structure is important to maintain for easy coding. An assistant is made of skills, intents, and slots. PocketPlay itself is a skill and the highest level of structure, you will not need to make another skill. Intents are how snips categorizes voice inputs. Each intent is a “job” such as a pilot or firefighter, all voice commands for a specific job should be put together in an intent. To make new jobs, simply create a new intent and train the intent with phrases that have keywords you want it to pick up. Use the already made pilot intent as an example. The more phrases that are used the more accurate the assistant becomes in picking up keywords.

The Snipsfile

The Snipsfile is the programmatic heart of PocketPlay and allows you to code what happens after a keyword is detected. It is structured as a json file but all work should be done below “intents:”.

**Intent:** This is the name of the intent from the assistant.

**Action:** This is where all the python code goes within the delimiters “{% %}”

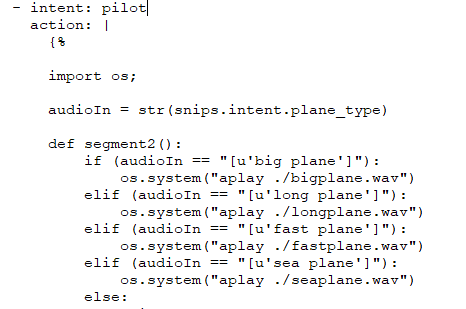
**audioIn:** The spoken keyword is contained within the object snips.intent.[slot\_type\_name]. It is accessed here by converting it to a String type which is in the format “[u’[keyword]’]”

**play\_Start Intent Structure**: play\_Start is the first intent that should be read when the user starts a session. It will identify which job the user wants, play the appropriate sound file, and start writing to a .txt file which saves the state of the story progress.

**Pilot Intent Structure**: This model should be used for all future intents (“jobs”) implemented. It divides the story into segments that play the appropriate sound file based on input and again save the state of the story into the .txt file to prevent incorrect phrases from being used

**Sound Files**: All sound files are located in usr/share/snips/assistant and must be there to be played. If you wish to play the sound files from elsewhere the code must be specified to their new location. The os system command “aplay” is only specified for use with .wav files

As seen below, an intent must first be defined with “- intent: [name of intent]” followed by “action: |”. Everything between the delimiters “{% %}” is python code.



**Caution:** The Snipsfile is very specific and will throw immediate errors for any formatting error. **No tabs must be used**; however, since Python is an indentation based language indentation is necessary so where tabs would normally be used, instead **exactly 4** **spaces** must be used. The first indentation level **must** start at the same indentation as the “ {% ”.