snips-app-helpers

Release 0.0.1

May 03, 2019

Contents

CHAPTER 1

Installation

At the command line:

```
pip install snips-app-helpers
```

CHAPTER 2

Usage

2.1 In Spec We Trust

One of the problem while devolopping a voice assistant using Snips is that we otfen endup having a missmatch between assistant possible intents and intents action. This can also happen that there is a slot name missmatch due to for example a console app that was developped by a developper and the action by another.

To fix this issue, we developped a tool that behave like this:

1. we expose **manually** for each action a "spec contract in yml" about which intents and slot the action will use, with a format as follow:

`your_app_dir/my_action_n1/spec.yml`

```
name: "My Action Code"

version: "0.0.2"
supported_snips_version: ["0.60.1", "0.62.3", ...]
updated_at: "12/04/2019"

slots:
   slot_name_A: "slot_type_startwith_1"
   slot_name_B: "slot_type_startwith_2"
   ...

coverage:
    intent1:
         [ ["slot_name_B", "+"], ["slot_name_A", "?"] ],
    intent2:
         [ ["slot_name_B", 1], ["slot_name_A", "{1,4}" ],
...
```

that coverage section is the one used to specify what to check you can either write it as shown upper or specify only the intent meaning all subcase like this:

```
coverage:
    intent1:
    intent2:
```

Note: this second method "mask" possible code coverage problem so we highly encourage you to use the first version (it will allow more fine grained analysis by our tools)

2.1.1 About the pattern quantifiers

The current implementation used of slot sequence pattern rule are only based on the quantity of each slot expected in the intent by the action code handler. And the following quantifiers follow as mutch as possible the regexp one. Here is the list of what is implemented now:

- "+" between 1 and n
- "*" between 0 and n
- "?" between 0 and 1
- "{1,10}" between 1 and 10
- 3 always 3

If the action code is not yours you obviously do not want to add the spec file in this 3rd party directory (who knows what will happen at next update). But you can create a spec file in one of your managed action code following the name convention: `your_app_dir/my_action_n1/{my_3rd_pary_action}.spec.yml` where *my_3rd_pary_action* is the 3rd party action code folder name

3. A cli match the concordence of both and report inconsistencies.

```
snips-app spec check --assistant_dir ... --actions_dir ...
```

(I invite you to alias snips-app to sap)

A typical report of the CLI looks like this:

```
Analysing spec for:
     assistant: /home/epi/open/projects/snips-app-helpers/tests/fixtures/assistant_1/
↪assistant.json
     app dir: /home/epi/open/projects/snips-app-helpers/tests/fixtures/actions_1

Detected spec:
        - @ Likhitha.Today/spec.yml applied to Likhitha.Today
        - @ Likhitha.Today/ozie.Calculations.spec.yml applied to ozie.Calculations
          ...

Intents do not seem to be covered by any action code:
        - currencyConverter
          ...
        Remarks:
                This might be due to missing spec in some action codes else you
                should take it seriously as no response at all will be given by your
                assistant to final user.

Some Intents seems to be hooked multiple times:
        - intent getCurrentTime in actions: ['Today', 'Music Player']
          ...
        Remarks:
```

(continues on next page)

(continued from previous page)

```
                While it might be legit do not forget that it means each time you
                trigger this intent n actions will be performed

Action waiting intent not in assistant:
        - MySuperFakeIntent from action: Music Player
        Remarks:
                This should not be a problem except that it consume resource with
                useless purpose

Missing spec for following actions:
        - Snips.Smart_Lights_-_Hue
          ...
```

2.2 Guess The Spec

As a first step if you already have a big assistant and do not want to add all specs manually, a system allow to guess what should be the good spec for each action code based on what intent and slots names are detected in the action code to use it you can directly use the following cli

```
snips-app spec auto-guess --spec_store ... --assistant_dir ... --actions_dir ...
```

2.3 The Spec Middleware

Once you have the specs defined as bellow you can use it to various purposes.

One of them is to match a action spec to an assistant spec, without modifying any of both. This is usefull in the case you want a console app and action to communicate but both beeing open 3rd party, or you develop only the action and dislike the interface. How is that possible ?

Thank to a middleware action code.

What it does ?

based on a routing file written in yml by the user, in the following form

`routing.yml`

```yaml
# routing table

"original_intent1":
   to: "routed_intent1"
   slots:
       original_slot_1: routed_slot_1
       original_slot_1: routed_slot_2
       ...
...
```

Then it you want to make your redirection work you need to install the action *src/actions/snips-app-middleware* with the *routing.yml* file in the same host and configure the config.ini to point to this one.

CHAPTER 3

Reference

3.1 snips_app_helpers

CHAPTER 4

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

4.1 Bug reports

When reporting a bug please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

4.2 Documentation improvements

snips-app-helpers could always use more documentation, whether as part of the official snips-app-helpers docs, in docstrings, or even on the web in blog posts, articles, and such.

4.3 Feature requests and feedback

The best way to send feedback is to file an issue at https://github.com/dreamermind/snips-app-helpers/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that code contributions are welcome :)

4.4 Development

To set up *snips-app-helpers* for local development:

1. Fork snips-app-helpers (look for the "Fork" button).
2. Clone your fork locally:

```
git clone git@github.com:your_name_here/snips-app-helpers.git
```

3. Create a branch for local development:

```
git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

4. When you're done making changes, run all the checks, doc builder and spell checker with tox one command:

```
tox
```

5. Commit your changes and push your branch to GitHub:

```
git add .
git commit -m "Your detailed description of your changes."
git push origin name-of-your-bugfix-or-feature
```

6. Submit a pull request through the GitHub website.

4.4.1 Pull Request Guidelines

If you need some code review or feedback while you're developing the code just make the pull request.

For merging, you should:

1. Include passing tests (run `tox`)[1].
2. Update documentation when there's new API, functionality etc.
3. Add a note to `CHANGELOG.rst` about the changes.
4. Add yourself to `AUTHORS.rst`.

4.4.2 Tips

To run a subset of tests:

```
tox -e envname -- pytest -k test_myfeature
```

To run all the test environments in *parallel* (you need to `pip install detox`):

```
detox
```

[1] If you don't have all the necessary python versions available locally you can rely on Travis - it will run the tests for each change you add in the pull request.
It will be slower though ...

4.5 Our Responsibilities

Project maintainers are responsible for clarifying the standards of acceptable behavior and are expected to take appropriate and fair corrective action in response to any instances of unacceptable behavior.

Project maintainers have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, or to ban temporarily or permanently any contributor for other behaviors that they deem inappropriate, threatening, offensive, or harmful.

4.6 Scope

This Code of Conduct applies both within project spaces and in public spaces when an individual is representing the project or its community. Examples of representing a project or community include using an official project e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event. Representation of a project may be further defined and clarified by project maintainers.

4.7 Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported by contacting the project team @snips.ai. All complaints will be reviewed and investigated and will result in a response that is deemed necessary and appropriate to the circumstances. The project team is obligated to maintain confidentiality with regard to the reporter of an incident. Further details of specific enforcement policies may be posted separately.

Project maintainers who do not follow or enforce the Code of Conduct in good faith may face temporary or permanent repercussions as determined by other members of the project's leadership.

4.8 Attribution

This Code of Conduct is adapted from the [Contributor Covenant][homepage], version 1.4, available at [http://contributor-covenant.org/version/1/4{]}{[}version]

[homepage]: http://contributor-covenant.org [version]: http://contributor-covenant.org/version/1/4/

CHAPTER 5

Authors

- Dreamermind - https://github.com/DreamerMind/

This is not an official Snips product, and is developed on my spare time so will I do my best to offer community better tools, but you accept use it at your own risks.

CHAPTER 6

Changelog

6.1 0.0.1 (2019-03-24)

- First release on PyPI.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

S

Index

S