
prakriya Documentation

Release 0.2.1

Dr. Dhaval Patel

Jan 15, 2019

Contents

1	prakriya	3
1.1	Features	3
1.2	Support	3
1.3	Credits	3
2	Installation	5
2.1	Stable release	5
2.2	From sources	5
3	Usage	7
3.1	prakriya	10
3.2	generate	11
4	Contributing	13
4.1	Types of Contributions	13
4.2	Get Started!	14
4.3	Pull Request Guidelines	15
4.4	Tips	15
5	Credits	17
5.1	Development Lead	17
5.2	Contributors	17
6	History	19
6.1	0.0.1 (2017-12-30)	19
6.2	0.0.2 (2018-01-01)	19
6.3	0.0.3 (2018-01-02)	19
6.4	0.0.4 (2018-01-03)	19
6.5	0.0.5 (2018-01-13)	19
6.6	0.0.6 (2018-01-16)	20
6.7	0.0.7 (2018-01-21)	20
6.8	0.1.0 (2018-12-17)	20
6.9	0.2.0 (2019-01-13)	20
6.10	0.2.1 (2019-01-14)	20
7	Indices and tables	21

Contents:

prakriya is a package to derive information about given Sanskrit verb form.

- Free software: GNU General Public License v3
- Documentation: <https://prakriya.readthedocs.io>.

1.1 Features

1. Provides step by step derivation based on Panini's grammar of Sanskrit.
2. Supports various transliteration as input and output.
3. Also provides a commandline tool.

1.2 Support

1. You can help development of code by testing and bug report.
2. You can contribute monetarily at the following link.

1.3 Credits

1. This package was created with [Cookiecutter](#) and the [audreyr/cookiecutter-pypackage](#) project template.
2. This package uses [indic-transliteration](#) package.

2.1 Stable release

To install prakriya, run this command in your terminal:

```
$ pip install prakriya
```

This is the preferred method to install prakriya, as it will always install the most recent stable release.

If you don't have [pip](#) installed, this [Python installation guide](#) can guide you through the process.

2.2 From sources

The sources for prakriya can be downloaded from the [Github repo](#).

You can either clone the public repository:

```
$ git clone git://github.com/drdhaval2785/python-prakriya
```

Or download the [tarball](#):

```
$ curl -OL https://github.com/drdhaval2785/python-prakriya/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```


Top-level package for prakriya.

class prakriya.Prakriya
Generate a prakriya class.

Example

To use prakriya in a project:

```
>>> from prakriya import Prakriya
>>> p = Prakriya()
```

If you are using the library the first time, be patient. This will take a long time, because the data file (30 MB) is being downloaded.

If you can spare around 600 MB space, It is highly recommended to decompress the tar.gz first time. Subsequent actions will be very fast. This is one time requirement. If the data is not decompressed, the code will read from tar.gz file every time. It introduces slowness to a great extent. So highly recommended to decompress.

```
>>> p.decompress()
```

Now you are ready to roll!

The generic format for usage is as follows:

```
>>> p.get_info(verbform, field)
```

verbform is mandatory. It is the verb form to be investigated. The input should be in SLP1 encoding.

field is optional.

Actual usage examples will be like the following.

```
>>> p.get_info('Bavati')
>>> p.get_info('Bavati', 'prakriya')
>>> p.get_info('Bavati', 'verb')
```

Valid values of field and expected output are as follows.

prakriya - Return step by step derivation.

verb - Return verb in Devanagari without accent marks.

verbaccent - Return the verb in Devanagari with accent marks.

lakara - Return the lakAra (tense / mood) in which this form is generated.

gana - Return the gaNa (class) of the verb.

meaning - Return meaning of the verb in SLP1 transliteration.

number - Return number of the verb in dhAtupATHa.

madhaviya - Return link to mAdhaviyadhAtuvRtti. <http://sanskrit.uohyd.ac.in/scl/dhaatupaatha> is the home page.

kshiratarangini - Return link to kSrataraGgiNI. <http://sanskrit.uohyd.ac.in/scl/dhaatupaatha> is the home page.

dhatupradipa - Return link to dhAtupradIpa. <http://sanskrit.uohyd.ac.in/scl/dhaatupaatha> is the home page.

jnu - Return link to JNU site for this verb form. <http://sanskrit.jnu.ac.in/tinanta/tinanta.jsp> is the home page.

uohyd - Return link to UoHyd site for this verb form. http://sanskrit.uohyd.ac.in/cgi-bin/scl/skt_gen/verb/verb_gen.cgi is the home page.

upasarga - Return upasarga, if any. Currently we do not support verb forms with upasargas.

padadecider_id - Return the rule number which decides whether the verb is parasmaipadI, AtmanepadI or ubhayapadI.

padadecider_sutra - Return the rule text which decides whether the verb is parasmaipadI, AtmanepadI or ubhayapadI.

it_id - Returns whether the verb is seT, aniT or veT, provided the form has iDAgama.

it_status - Returns whether the verb form has iDAgama or not. seT, veT, aniT are the output.

it_sutra - Returns rule number if iDAgama is caused by some special rule.

purusha - Returns the purusha of the given verb form.

vachana - Returns the vacana of the given verb form.

transliteration

If you want to set the input or output transliteration, follow these steps.

```
>>> from prakriya import Prakriya
>>> p = Prakriya()
>>> p.inputTranslit('hk') # Customize 'hk'
>>> p.outputTranslit('devanagari') # Customize 'devanagari'
>>> p.get_info('bhavati') # Input in HK and output in Devanagari.
>>> p.inputTranslit('devanagari')
>>> p.outputTranslit('iast')
>>> p.get_info('') # Input in Devanagari and output in IAST.
```

Valid transliterations are slp1, itrans, hk, iast, devanagari, wx, bengali, gujarati, gurmukhi, kannada, malayalam, oriya and telugu. They can be used both as input transliteration and output transliteration.

decompress()

Decompress the tar file if user asks for it.

get_data(verbform, tar, inTran='slp1', outTran='slp1')

Get whole data from the json file for given verb form.

get_info(verbform, field='prakriya')

Return the data requested by user.

inputTranslit(tran)

Set input transliteration.

outputTranslit(tran)

Set output transliteration.

class prakriya.VerbFormGenerator

Return the verb form for given verb, tense, purusha-vachana or suffix.

Example

To get verb form for given verb, tense, suffix / (purusha and vachana) in a project:

```
>>> from prakriya import VerbFormGenerator
>>> g = VerbFormGenerator()
```

There are four ways to get verb forms for given verb.

```
>>> g.getforms(inputverb, lakara='', purusha='', vachana='')
>>> g.getforms(inputverb, lakara='', suffix='')
>>> g[verb, tense, purusha, vachana]
>>> g[verb, tense, suffix]
# Examples of four formats are as follows. Default input transliteration_
↳ is SLP1.
>>> g.getforms('BU', 'law', 'praTama', 'bahu')
>>> g.getforms('BU', 'law', 'Ji')
>>> g['BU', 'law', 'praTama', 'eka']
>>> g['BU', 'law', 'tip']
```

`__getitem__` method is discouraged. Will be deprecated in later versions.

For using transliterations in VerbFormGenerator class, use as below.

```
>>> from prakriya import VerbFormGenerator
>>> g = VerbFormGenerator()
>>> g.inputTranslit('hk') # Customize 'hk'
>>> g.outputTranslit('devanagari') # Customize 'devanagari'
>>> g.getforms('bhU', 'laT', 'prathama', 'bahu') # Input in HK and output in_
↳ Devanagari.
```

Valid transliterations are slp1, itrans, hk, iast, devanagari, wx, bengali, gujarati, gurmukhi, kannada, malayalam, oriya and telugu. They can be used both as input transliteration and output transliteration.

inputTranslit(tran)

Set input transliteration.

outputTranslit(tran)

Set output transliteration.

3.1 prakriya

Console script to get derivation and other information for given verb form.

```
$ prakriya [OPTIONS] VERBFORM [FIELD]
```

Valid values of FIELD and expected output are as follows. prakriya - Return step by step derivation.

verb - Return verb in Devanagari without accent marks.

verbaccent - Return the verb in Devanagari with accent marks.

“lakara” - Return the lakAra (tense / mood) in which this form is generated.

purusha - Returns purusha of the given verb form.

vachana - Returns vachana of the given verb form.

gana - Return the gaNa (class) of the verb.

meaning - Return meaning of the verb in SLP1 transliteration.

number - Return number of the verb in dhAtupATha.

madhaviya - Return link to mAdhaviyadhAtuvRtti. <http://sanskrit.uohyd.ac.in/scl/dhaatupaatha> is the home page.

kshiratarangini - Return link to kSIrataragGiNI. <http://sanskrit.uohyd.ac.in/scl/dhaatupaatha> is the home page.

dhatupradipa - Return link to dhAtupradIpa. <http://sanskrit.uohyd.ac.in/scl/dhaatupaatha> is the home page.

jnu - Return link to JNU site for this verb form. <http://sanskrit.jnu.ac.in/tinanta/tinanta.jsp> is the home page.

uohyd - Return link to UoHyd site for this verb form. http://sanskrit.uohyd.ac.in/cgi-bin/scl/skt_gen/verb/verb_gen.cgi is the home page.

upasarga - Return upasarga, if any. Currently we do not support verb forms with upasargas.

padadecider_id - Return the rule number which decides whether the verb is parasmaipadI, AtmanepadI or ubhayapadI.

padadecider_sutra - Return the rule text which decides whether the verb is parasmaipadI, AtmanepadI or ubhayapadI.

it_id - Returns whether the verb is seT, aniT or veT, provided the form has iDAgama.

it_status - Returns whether the verb form has iDAgama or not. seT, veT, aniT are the output.

it_sutra - Returns rule number if iDAgama is caused by some special rule.

prakriya [OPTIONS] VERBFORM [FIELD]

Options

--intran <intran>

--outtran <outtran>

Arguments

VERBFORM

Required argument

FIELD

Optional argument

3.2 generate

Console script to get verb form for given verb, lakara, purusha and vachana.

```
$ generate [OPTIONS] VERB LAKARA PURUSHA VACHANA
```

LAKARA can take law, liw, luw, lfw, low, laN, viDiliN, ASIrliN, luN, lfn values.

PURUSHA can take praTama, maDyama, uttama values.

VACHANA can take eka, dvi, bahu values.

```
generate [OPTIONS] VERB LAKARA PURUSHA VACHANA
```

Options

```
--intran <intran>
```

```
--outtran <outtran>
```

Arguments

VERB

Required argument

LAKARA

Required argument

PURUSHA

Required argument

VACHANA

Required argument

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given. You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at <https://github.com/drdhaval2785/python-prakriya/issues>.

If you are 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.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” and “help wanted” is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “enhancement” and “help wanted” is open to whoever wants to implement it.

4.1.4 Write Documentation

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

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/drdhaval2785/python-prakriya/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 contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here's how to set up *prakriya* for local development.

1. Fork the *python-prakriya* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:drdhaval2785/python-prakriya.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv prakriya
$ cd prakriya/
$ python setup.py develop
```

4. Create a branch for local development:

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

Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 prakriya tests
$ tox
```

To get flake8 and tox, just pip install them into your virtualenv.

6. 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
```

7. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.7, 3.3, 3.4, 3.5, 3.6 and for PyPy. Check https://travis-ci.org/drdhaval2785/python-prakriya/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_prakriya
```


5.1 Development Lead

- Dr. Dhaval Patel <drdhaval2785@gmail.com>

5.2 Contributors

None yet. Why not be the first?

6.1 0.0.1 (2017-12-30)

- First release on PyPI.

6.2 0.0.2 (2018-01-01)

Broken release. Kindly update to v0.0.3 immediately.

6.3 0.0.3 (2018-01-02)

A fix for broken v0.0.2 release. jsonindex.json file which was crucial and missed in v0.0.2, added back.

6.4 0.0.4 (2018-01-03)

1. Added transliteration support. See #5 .
2. Adding data file in system specific locations like APPDATA. See #6 .
3. Made code python 2.7, 3.3, 3.4, 3.5, 3.6 compliant.
4. Corrected all errors / smells identified by landscape.io.

6.5 0.0.5 (2018-01-13)

1. Added verb form generation facility. See <https://github.com/drdhaval2785/prakriya/issues/39>.

6.6 0.0.6 (2018-01-16)

1. Added CLI functionality ‘generate’. See <https://github.com/drdhaval2785/python-prakriya/issues/12>.
2. Guess the correct verb e.g. eD -> eDa~. See <https://github.com/drdhaval2785/python-prakriya/issues/13>.
3. All data files moved to appdata folder. See <https://github.com/drdhaval2785/python-prakriya/issues/14>.
4. Memoize data read from JSONs for speedup. See <https://github.com/drdhaval2785/python-prakriya/issues/15>.
5. Removed unnecessary duplicate loading of JSONs. See <https://github.com/drdhaval2785/python-prakriya/issues/11>.

6.7 0.0.7 (2018-01-21)

1. Changed the output format of Generate class. See <https://github.com/drdhaval2785/prakriya/issues/37#issuecomment-359224797>.

6.8 0.1.0 (2018-12-17)

1. VerbFormGenerator now uses methods instead of getitem. See <https://github.com/drdhaval2785/python-prakriya/issues/36>.

6.9 0.2.0 (2019-01-13)

1. Prakriya now uses method instead of getitem. See <https://github.com/drdhaval2785/python-prakriya/issues/75>.

6.10 0.2.1 (2019-01-14)

1. Documentation of commandline (CLI) tools added in documentation.

CHAPTER 7

Indices and tables

- `genindex`
- `modindex`
- `search`

p

prakriya, [7](#)

Symbols

–intran <intran>
 generate command line option, 11
 prakriya command line option, 10
–outtran <outtran>
 generate command line option, 11
 prakriya command line option, 10

D

decompress() (prakriya.Prakriya method), 9

F

FIELD
 prakriya command line option, 11

G

generate command line option
 –intran <intran>, 11
 –outtran <outtran>, 11
 LAKARA, 11
 PURUSHA, 11
 VACHANA, 11
 VERB, 11
get_data() (prakriya.Prakriya method), 9
get_info() (prakriya.Prakriya method), 9

I

inputTranslit() (prakriya.Prakriya method), 9
inputTranslit() (prakriya.VerbFormGenerator method), 9

L

LAKARA
 generate command line option, 11

O

outputTranslit() (prakriya.Prakriya method), 9
outputTranslit() (prakriya.VerbFormGenerator method), 9

P

Prakriya (class in prakriya), 7
prakriya (module), 7
prakriya command line option
 –intran <intran>, 10
 –outtran <outtran>, 10
 FIELD, 11
 VERBFORM, 11
PURUSHA
 generate command line option, 11

V

VACHANA
 generate command line option, 11
VERB
 generate command line option, 11
VERBFORM
 prakriya command line option, 11
VerbFormGenerator (class in prakriya), 9