Heppi Documentation

Release 0.1.0

Yacine Haddad

February 14, 2016

Contents

Contents:

CHAPTER 1

Heppi

A High Energy Physics Plotting Interface

- Free software: ISC license
- Documentation: https://heppi.readthedocs.org.

1.1 How to run

1.1.1 Produce a stacked plots

- To run `heepi` plotmaker you have to run the script `plot`
- you can print the options of the script by typing `.\plot --help`
- you have to combine the trees using `rootmerge.py` script
- the commend I'm using is the follwing:
- if you want to print one variable in the plotcard you can replace the option `--all` by `--variable` or just `-v` followed by the name of the variable.

example : .. code-block:

```
./plot -s /dir/to/merged/trees --load plotcard.json -v var1
```

1.1.2 Write a valid plotcard ?

- Produce the plotcard using a processe.json files and input root file.
- The tree name must be specified
- the `*` will be replaced automatically by the remaining name of the tree found in the `VBFMVADumper` directory.
- This is for the use of .. 'flashgg': https://github.com/cms-analysis/flashgg type dumper trees only, a more standard version will be pushed soon

1.2 installation

- Run the setup script to install the dependencies: `python setup.py develop --user` or run : `pip install --user jsmin termcolor progressbar jsonmerge`, both ways works on lxplus.

- ROOT env must be set, I recomend to setup CMSSW env before runnning the previous commands

1.3 Credits

This package was created with Cookiecutter and the audreyr/cookiecutter-pypackage project template.

CHAPTER 2

Installation

At the command line:

```
$ easy_install heppi
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv heppi
$ pip install heppi
```

CHAPTER 3

Usage

To use Heppi in a project:

```
import heppi
```

CHAPTER 4

Contributing

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/yhaddad/heppi/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" is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "feature" is open to whoever wants to implement it.

4.1.4 Write Documentation

heppi could always use more documentation, whether as part of the official heppi 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/yhaddad/heppi/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 *heppi* for local development.

1. Fork the *heppi* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/heppi.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 heppi
$ cd heppi/
$ 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 heppi tests
$ python setup.py test
$ 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 2.6, 2.7, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/yhaddad/heppi/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_heppi
```

CHAPTER 5

Credits

5.1 Development Lead

- Yacine Haddad <yhaddad@cern.ch>

5.2 Contributors

None yet. Why not be the first?

CHAPTER 6

History

6.1 0.1.0 (2015-12-13)

- First release on PyPI.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search