BMicro Documentation

Release 0.1.3

Paul Müller

Apr 30, 2021

CONTENTS:

This is BMicro, a graphical user interface for Brillouin microscopy data evaluation. This is the documentation of bmlab version 0.1.3.

CHAPTER

ONE

GETTING STARTED

1.1 Installation

To install BMicro, use one of the following methods:

- **from PyPI:** `pip install bmicro`
- **from sources:** `pip install .`

1.2 Citing BMicro

If you use BMicro in a scientific publication, please cite it with:

> BMicro developers (2021), BMicro version X.X.X: Python library for the post-measurement analysis of real-time deformability cytometry data sets [Software]. Available at https://github.com/BrillouinMicroscopy/BMicro.

CHAPTER

TWO

CODE REFERENCE

TODO

CHAPTER

THREE

DEVELOPMENT

This section gives on overview about everything you need to know if you wish to contribute to bmlab or BMicro.

3.1 Development workflow

We use GitHub projects to manage the development workflow of bmlab and BMicro. The main development project board is Brillouin Evaluation in Python.

The development is split into "User Stories", each of which is a collection of issues (identified via titles in the issues). The current work in progress (WIP) branch is named according to the currently active user story (e.g. *1-smoke-test*). Issues that are not part of the current user story should still be addressed in the current WIP branch.

Once you wish to address an issue, drag it from the "Open" or "Ready" column of the project board to the "In progress" column. Once you finished working on an issue, drag it to the "Done" column, but don't close it yet (It should be discussed first in the dev meeting).

Notes:

- Please write test functions and keep code coverage above 90%.
- Please make sure to always edit the changelog for BMicro or bmlab.
- Please try to always pull with rebase

  ```
  git pull --rebase
  ```

  instead of

  ```
  git pull
  ```

  to prevent confusions in the commit history.

3.2 Documentation

It is always helpful to have code examples and thorough descriptions in a documentation. We use sphinx-autodoc for the *code reference*, which means that the docstrings of your functions and classes are automatically rendered. Please make sure that this is working properly - go to the `docs` directory and execute:

```
pip install -r requirements.txt
sphinx-build . _build
```

This will create a a file `_build/index.html` which you can open in your favorite browser. This also applies to bmlab.

3.3 Tests

We try to adhere to test-driven development. Please always write test functions for your code. Make sure you have the required packages installed:

```
pip install -r tests/requirements.txt
```

You can run all tests via

```
python -m pytest tests
```

To check for code coverage, make sure the *coverage* Python package is installed and run

```
coverage run --source="bmicro" -m pytest tests
coverage report
```

3.4 Making a new release

The release process of BMicro is completely automated. All you need to know is that you have to create an incremental tag:

```
git tag -a "0.1.3"
# or (if you have set up PGP)
git tag -s "0.1.3"
# and finally
git push --tags
```

For more information on how automatic deployment to PyPI works, please read on.

3.5 Continuous integration

The following things are automated:

- pytest and flake8 on Linux, macOS, and Windows via GitHub Actions: https://github.com/BrillouinMicroscopy/BMicro/actions?query=workflow%3AChecks

  You should always check that all checks pass before you merge a pull request (A green state on your local machine does not mean a global green state).

- automatic deployment to PyPI on tag creation via GitHub Actions: https://github.com/BrillouinMicroscopy/BMicro/actions?query=workflow%3A%22Release+to+PyPI%22

  Paul Müller created the BMicro package on PyPI and gave the user `ci_bm` permission to upload new releases. The password for this user is an organization secret.

- documentation is built automatically (for all tags and for the latest commit to the main branch) on readthedocs: https://readthedocs.org/projects/BMicro/builds/

- coverage statistics are done with codecov: https://codecov.io/gh/BrillouinMicroscopy/BMicro

  Please try stay above 90% coverage.

Badges for all of these CI tasks are in the main `README.rst` file.

CHAPTER

FOUR

CHANGELOG

List of changes in-between bmlab releases.

4.1 version 0.1.3

- setup: use “pytest” command instead of deprecated “setup.py test”
- ui: add app icon (#9)
- ui: data tab (#11)
- build: add Windows and macOS build pipeline (#1)

4.2 version 0.1.2

- CI automation

4.3 version 0.1.1

- Test CI automation

4.4 version 0.1.0

- dummy release

CHAPTER

FIVE

BILBLIOGRAPHY

CHAPTER

SIX

INDICES AND TABLES

- genindex
- modindex
- search