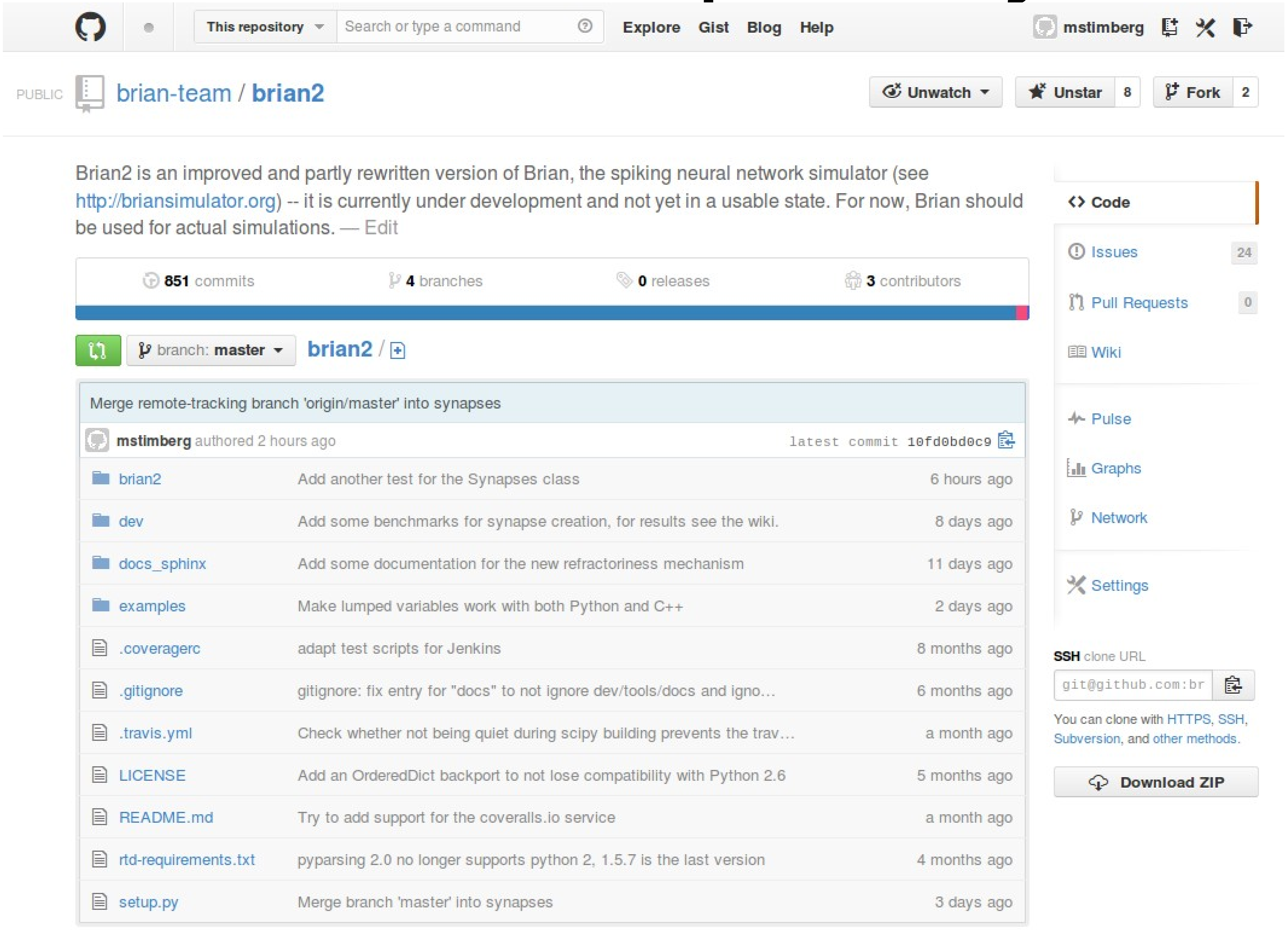
Contributing to Brian²

Marcel Stimberg, ENS Paris
marcel.stimberg@ens.fr

Github repository



This repository | Search or type a command | Explore Gist Blog Help | mstimberg

PUBLIC brian-team / brian2 — Unwatch | Unstar 8 | Fork 2

Brian2 is an improved and partly rewritten version of Brian, the spiking neural network simulator (see http://briansimulator.org) -- it is currently under development and not yet in a usable state. For now, Brian should be used for actual simulations. — Edit

851 commits | 4 branches | 0 releases | 3 contributors

branch: master | brian2 /

Merge remote-tracking branch 'origin/master' into synapses

mstimberg authored 2 hours ago — latest commit 10fd0bd0c9

brian2	Add another test for the Synapses class	6 hours ago
dev	Add some benchmarks for synapse creation, for results see the wiki.	8 days ago
docs_sphinx	Add some documentation for the new refractoriness mechanism	11 days ago
examples	Make lumped variables work with both Python and C++	2 days ago
.coveragerc	adapt test scripts for Jenkins	8 months ago
.gitignore	gitignore: fix entry for "docs" to not ignore dev/tools/docs and igno…	6 months ago
.travis.yml	Check whether not being quiet during scipy building prevents the trav…	a month ago
LICENSE	Add an OrderedDict backport to not lose compatibility with Python 2.6	5 months ago
README.md	Try to add support for the coveralls.io service	a month ago
rtd-requirements.txt	pyparsing 2.0 no longer supports python 2, 1.5.7 is the last version	4 months ago
setup.py	Merge branch 'master' into synapses	3 days ago

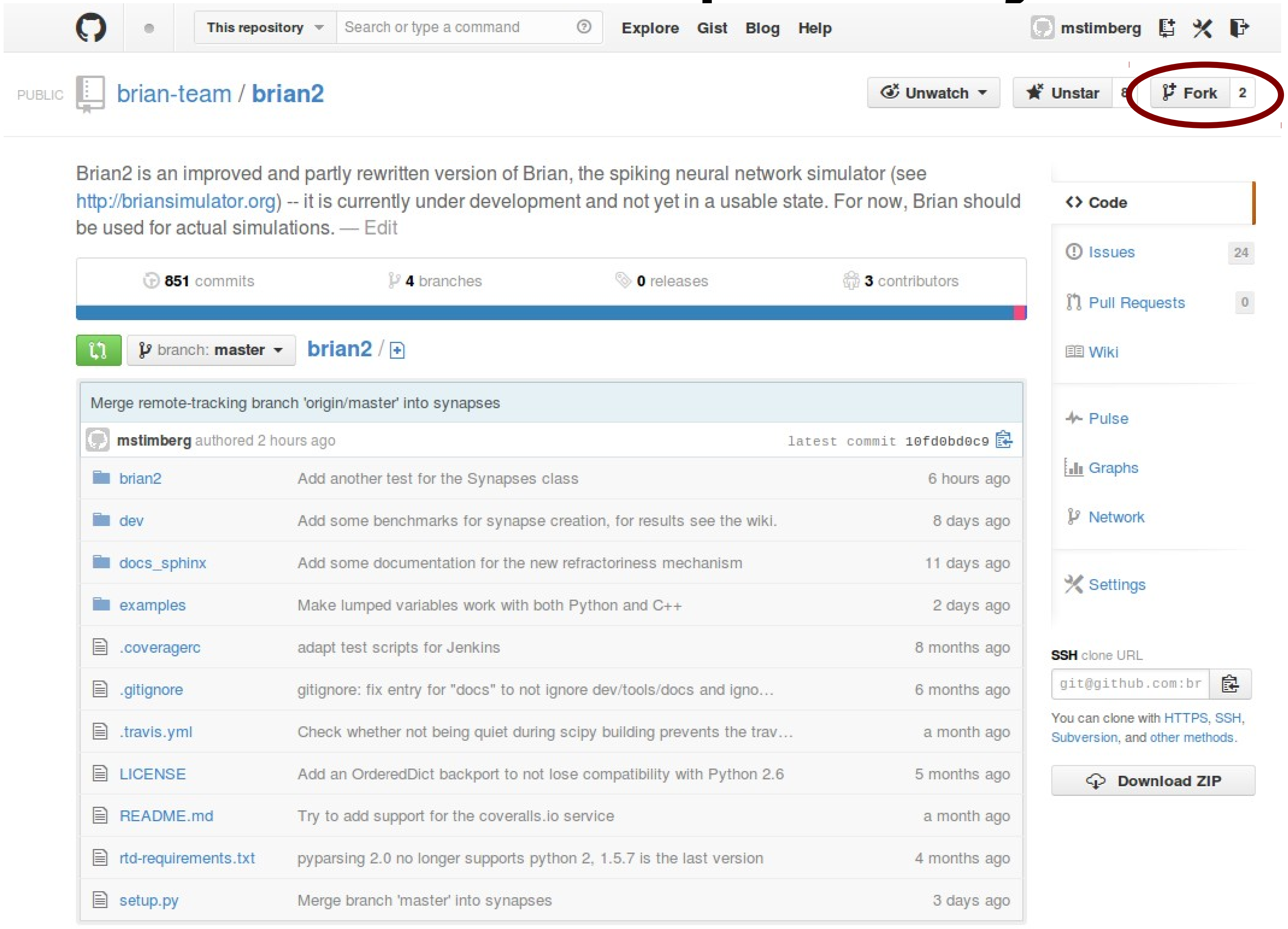
SSH clone URL: git@github.com:br

You can clone with HTTPS, SSH, Subversion, and other methods.

Download ZIP

https://github.com/brian-team/brian2

Github repository



This repository | Search or type a command | Explore | Gist | Blog | Help | mstimberg

PUBLIC brian-team / brian2 | Unwatch | Unstar | Fork 2

Brian2 is an improved and partly rewritten version of Brian, the spiking neural network simulator (see http://briansimulator.org) -- it is currently under development and not yet in a usable state. For now, Brian should be used for actual simulations. — Edit

851 commits | 4 branches | 0 releases | 3 contributors

branch: master | brian2 / +

Merge remote-tracking branch 'origin/master' into synapses

mstimberg authored 2 hours ago | latest commit 10fd0bd0c9

brian2	Add another test for the Synapses class	6 hours ago
dev	Add some benchmarks for synapse creation, for results see the wiki.	8 days ago
docs_sphinx	Add some documentation for the new refractoriness mechanism	11 days ago
examples	Make lumped variables work with both Python and C++	2 days ago
.coveragerc	adapt test scripts for Jenkins	8 months ago
.gitignore	gitignore: fix entry for "docs" to not ignore dev/tools/docs and igno…	6 months ago
.travis.yml	Check whether not being quiet during scipy building prevents the trav…	a month ago
LICENSE	Add an OrderedDict backport to not lose compatibility with Python 2.6	5 months ago
README.md	Try to add support for the coveralls.io service	a month ago
rtd-requirements.txt	pyparsing 2.0 no longer supports python 2, 1.5.7 is the last version	4 months ago
setup.py	Merge branch 'master' into synapses	3 days ago

SSH clone URL: git@github.com:br

You can clone with HTTPS, SSH, Subversion, and other methods.

Download ZIP

https://github.com/brian-team/brian2

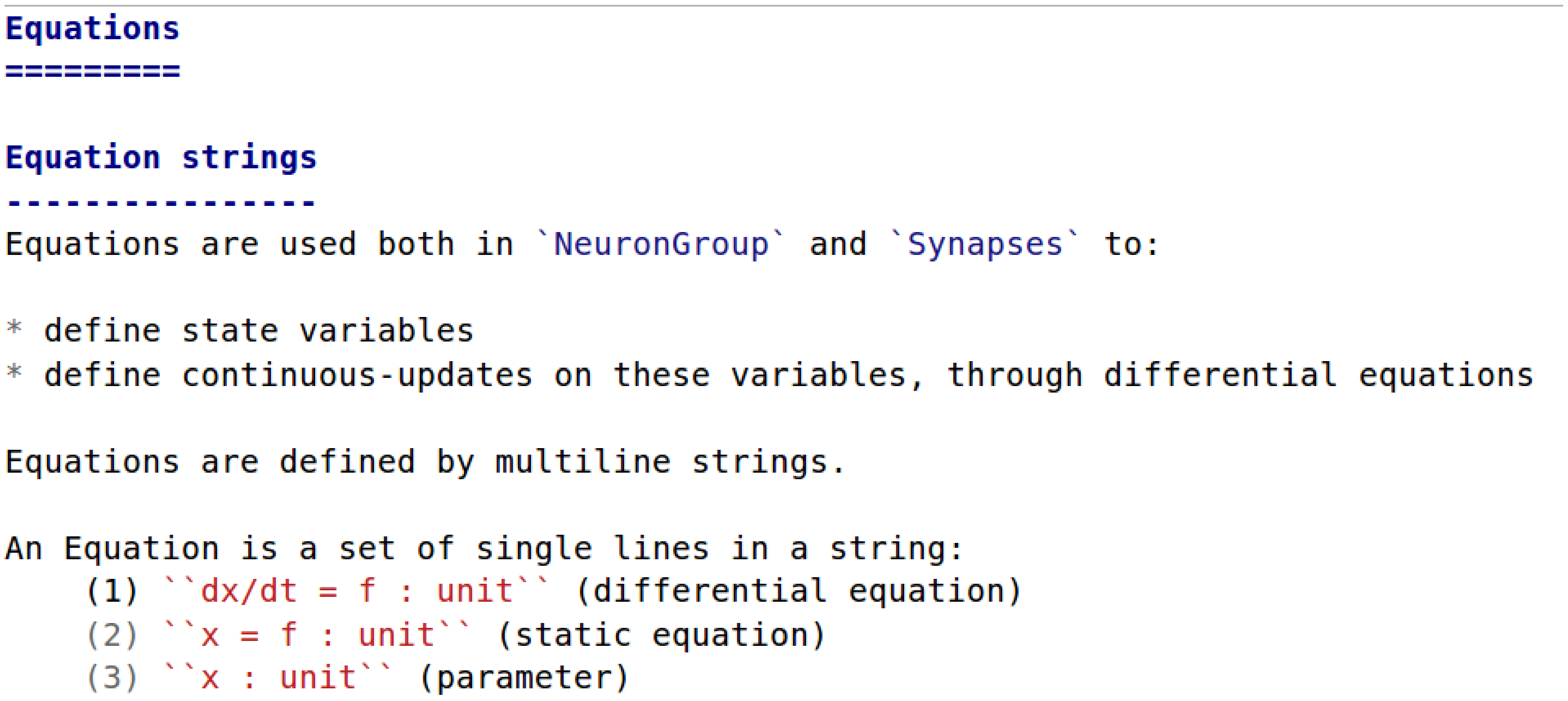
Development workflow

- File an issue
- Fork the project
- Create a branch (recommended)
- Create a pull request
- Refine the code until the reviewer is happy
- The reviewer merges the change into Brian
- Enjoy the eternal gratitude of the Brian community

https://brian2.readthedocs.org/en/latest/developer/guidelines/workflow.html

Brian documentation

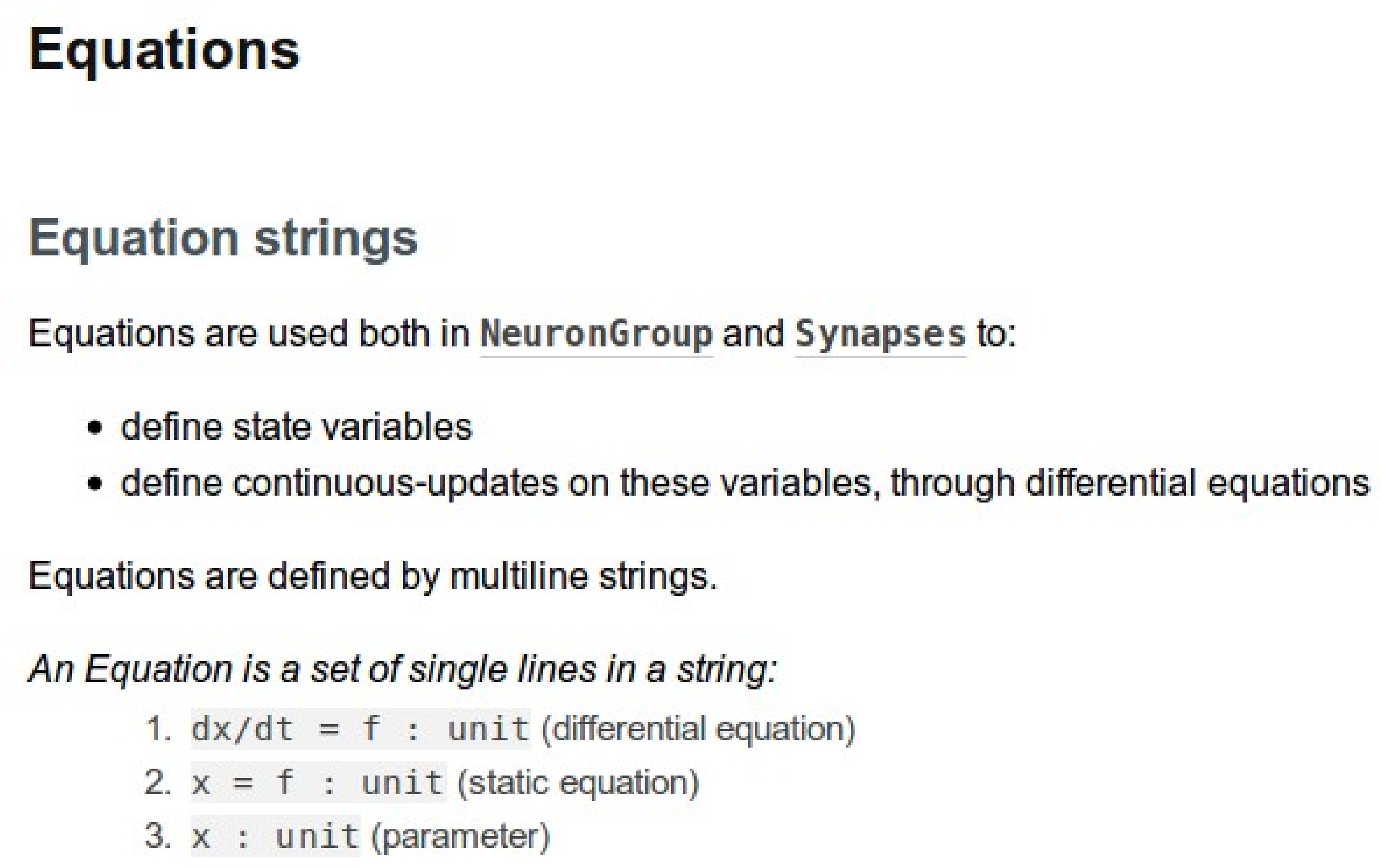
Restructured text:



```
Equations
=========

Equation strings
----------------
Equations are used both in `NeuronGroup` and `Synapses` to:

* define state variables
* define continuous-updates on these variables, through differential equations

Equations are defined by multiline strings.

An Equation is a set of single lines in a string:
    (1) ``dx/dt = f : unit`` (differential equation)
    (2) ``x = f : unit`` (static equation)
    (3) ``x : unit`` (parameter)
```

Brian documentation

Rendered Restructured text:



Equations

Equation strings

Equations are used both in `NeuronGroup` and `Synapses` to:

- define state variables
- define continuous-updates on these variables, through differential equations

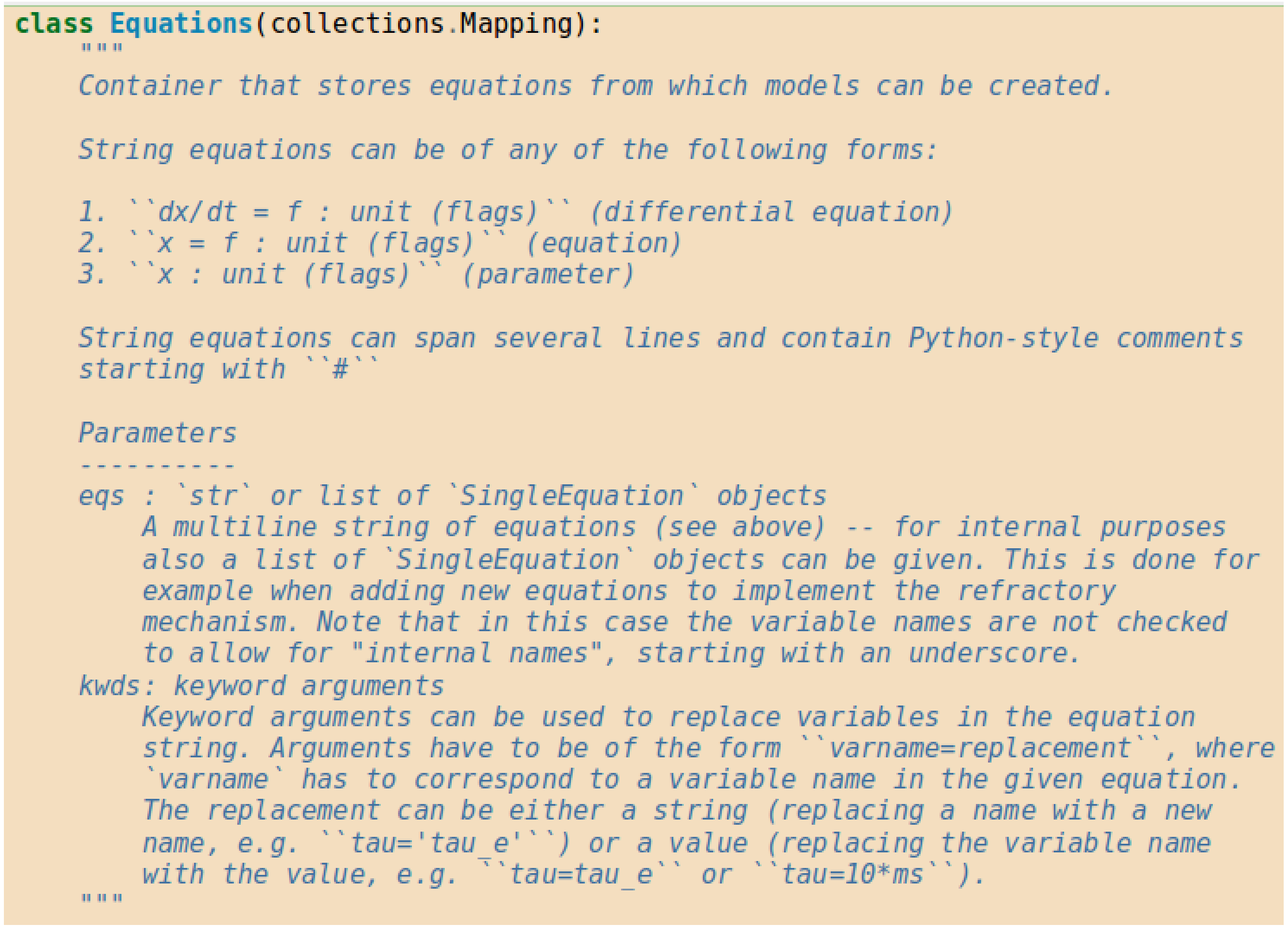
Equations are defined by multiline strings.

An Equation is a set of single lines in a string:

1. `dx/dt = f : unit` (differential equation)
2. `x = f : unit` (static equation)
3. `x : unit` (parameter)

Brian documentation

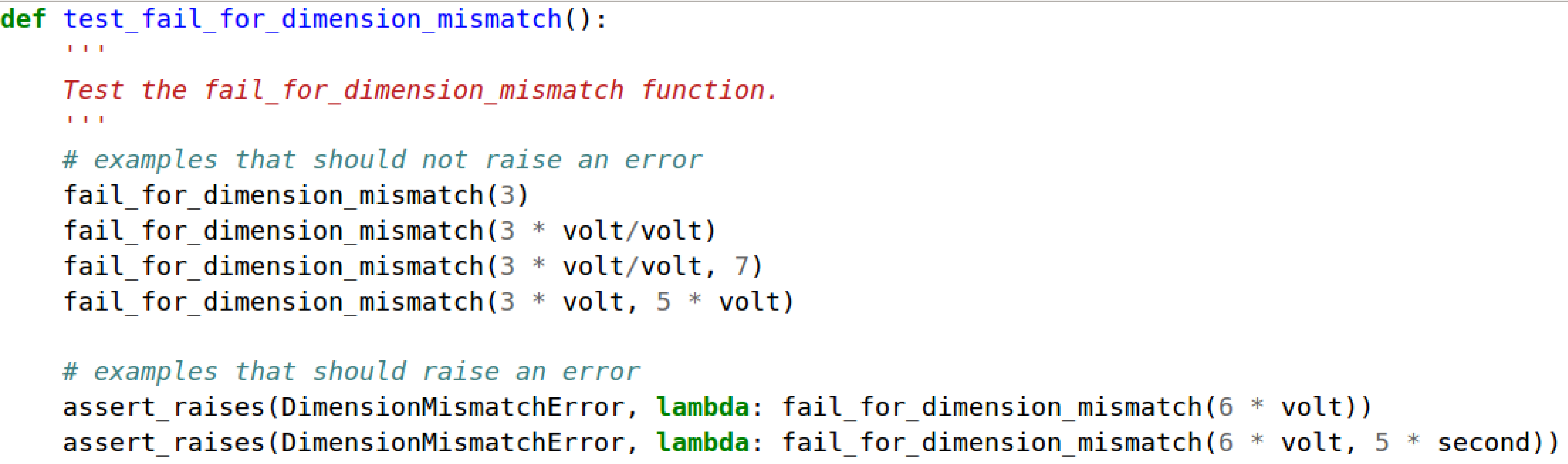
Docstrings using numpydoc style



```
class Equations(collections.Mapping):
    """
    Container that stores equations from which models can be created.

    String equations can be of any of the following forms:

    1. ``dx/dt = f : unit (flags)`` (differential equation)
    2. ``x = f : unit (flags)`` (equation)
    3. ``x : unit (flags)`` (parameter)

    String equations can span several lines and contain Python-style comments
    starting with ``#``

    Parameters
    ----------
    eqs : `str` or list of `SingleEquation` objects
        A multiline string of equations (see above) -- for internal purposes
        also a list of `SingleEquation` objects can be given. This is done for
        example when adding new equations to implement the refractory
        mechanism. Note that in this case the variable names are not checked
        to allow for "internal names", starting with an underscore.
    kwds: keyword arguments
        Keyword arguments can be used to replace variables in the equation
        string. Arguments have to be of the form ``varname=replacement``, where
        `varname` has to correspond to a variable name in the given equation.
        The replacement can be either a string (replacing a name with a new
        name, e.g. ``tau='tau_e'``) or a value (replacing the variable name
        with the value, e.g. ``tau=tau_e`` or ``tau=10*ms``).
    """
```

Writing tests

- Tests are automatically collected from `brian2.test` (filenames starting with `test`, functions starting with `test`)
- Tests should cover a single function/feature, use `assert` statements



```python
def test_fail_for_dimension_mismatch():
    '''
    Test the fail_for_dimension_mismatch function.
    '''
    # examples that should not raise an error
    fail_for_dimension_mismatch(3)
    fail_for_dimension_mismatch(3 * volt/volt)
    fail_for_dimension_mismatch(3 * volt/volt, 7)
    fail_for_dimension_mismatch(3 * volt, 5 * volt)

    # examples that should raise an error
    assert_raises(DimensionMismatchError, lambda: fail_for_dimension_mismatch(6 * volt))
    assert_raises(DimensionMismatchError, lambda: fail_for_dimension_mismatch(6 * volt, 5 * second))
```

Tools we use

- github.com (source code repository, issues, wiki)
- travis-ci.org (continuous integration testing)
- coveralls.io (test coverage)
- readthedocs.org (documentation hosting/building)

Travis CI

Read the Docs

Create, host, and browse documentation.

Where to contribute

- Extending Brian: State updaters, new code generation languages, …
- Fixing bugs (including "easy fixes": mistakes in documentation, better error messages)
- Writing tests to increase test coverage
- Writing and submitting examples
- Reporting bugs