Toga-Chart Documentation

Release 0.2.0

Russell Keith-Magee

Nov 03, 2023

CONTENTS

Toga Chart is Matplotlib chart widget for Toga.

CHAPTER

ONE

TABLE OF CONTENTS

1.1 Tutorial

Get started with a hands-on introduction for beginners

1.2 How-to guides

Guides and recipes for common problems and tasks, including how to contribute

1.3 Background

Explanation and discussion of key topics and concepts

1.4 Reference

Technical reference - commands, modules, classes, methods

CHAPTER

TWO

COMMUNITY

Toga Chart is part of the BeeWare suite. You can talk to the community through:

- @pybeeware on Twitter
- beeware/general on Gitter

2.1 Tutorial

Coming soon…

2.2 How-to guides

How-to guides are recipes that take the user through steps in key subjects. They are more advanced than tutorials and assume a lot more about what the user already knows than tutorials do, and unlike documents in the tutorial they can stand alone.

2.2.1 Contributing code to Toga-Chart

If you experience problems with Toga-Chart, log them on GitHub. If you want to contribute code, please fork the code and submit a pull request.

Setting up your development environment

The recommended way of setting up your development environment for Toga-Chart is to use a virtual environment, install the required dependencies and start coding:

macOS

```
$ git clone https://github.com/beeware/toga-chart.git
$ cd toga-chart
$ python3 -m venv venv
$ . venv/bin/activate
```

Linux

```
$ git clone https://github.com/beeware/toga-chart.git
$ cd toga-chart
$ python3 -m venv venv
$ . venv/bin/activate
```

Windows

```
C:\...>git clone https://github.com/beeware/toga-chart.git
C:\...>cd toga-chart
C:\...>py -m venv venv
C:\...>venv\Scripts\activate
```

To install all the development version of Toga-Chart, along with all it's requirements, run the following commands within your virtual environment:

macOS

```
$ (venv) pip install -e .
```

Linux

```
$ (venv) pip install -e .
```

Windows

```
C:\...>pip install -e .
```

Now you are ready to start hacking! Have fun!

Toga-Chart uses PyTest for its own test suite. It uses tox to manage the testing process. To set up a testing environment and run the full test suite:

macOS

```
$ (venv) pip install tox
$ (venv) tox
```

Linux

```
$ (venv) pip install tox
$ (venv) tox
```

Windows

```
C:\...>pip install tox
C:\...>tox
```

By default this will run the test suite multiple times, once on each Python version supported by Toga-Chart, as well as running some pre-commit checks of code style and validity. This can take a while, so if you want to speed up the process while developing, you can run the tests on one Python version only:

macOS

```
(venv) $ tox -e py
```

Linux

```
(venv) $ tox -e py
```

Windows

```
C:\...>tox -e py
```

Or, to run using a specific version of Python:

macOS

```
(venv) $ tox -e py
```

Linux

```
(venv) $ tox -e py
```

Windows

```
C:\...>tox -e py
```

substituting the version number that you want to target. You can also specify one of the pre-commit checks *flake8*, *docs* or *package* to check code formatting, documentation syntax and packaging metadata, respectively.

2.2.2 Contributing to the documentation

Here are some tips for working on this documentation. You're welcome to add more and help us out!

First of all, you should check the Restructured Text (reST) and Sphinx CheatSheet to learn how to write your .rst file.

Create a .rst file

Look at the structure and choose the best category to put your .rst file. Make sure that it is referenced in the index of the corresponding category, so it will show on in the documentation. If you have no idea how to do this, study the other index files for clues.

Build documentation locally

To build the documentation locally, *set up a development environment*, and run:

macOS

```
(venv) $ tox -e docs
```

Linux

```
(venv) $ tox -e docs
```

Windows

```
C:\...>tox -e docs
```

The output of the file should be in the `build/sphinx/html` folder. If there are any markup problems, they'll raise an error.

2.2.3 Internal How-to guides

These guides are for the maintainers of the Briefcase project, documenting internal project procedures.

How to cut a toga-chart release

The release infrastructure for toga-chart is semi-automated, using GitHub Actions to formally publish releases.

This guide assumes that you have an `upstream` remote configured on your local clone of the toga-chart repository, pointing at the official repository. If all you have is a checkout of a personal fork of the toga-chart repository, you can configure that checkout by running:

```
$ git remote add upstream https://github.com/beeware/toga-chart.git
```

The procedure for cutting a new release is as follows:

1. Check the contents of the upstream repository's main branch:

   ```
   $ git fetch upstream
   $ git checkout --detach upstream/main
   ```

   Check that the HEAD of release now matches upstream/main.

2. Make sure the branch is ready for release. Ensure that:

   1. The version number has been bumped.
   2. The release notes are up to date. If they are, the changes directory should be empty, except for the `template.rst` file.

   These two changes (the version bump and release notes update) should go through the normal pull request and review process. They should generally comprise the last PR merged before the release occurs.

   If the version number *hasn't* been updated, or `changes` directory *isn't* empty, you need to create a PR (using the normal development process) that contains these changes. Run:

   ```
   $ tox -e towncrier -- --draft
   ```

   to review the release notes that will be included, and then:

   ```
   $ tox -e towncrier
   ```

   to generate the updated release notes. Submit the PR; once it's been reviewed and merged, you can restart the release process from step 1.

3. Tag the release, and push the tag upstream:

   ```
   $ git tag v1.2.3
   $ git push upstream main
   $ git push upstream v1.2.3
   ```

4. Pushing the tag will start a workflow to create a draft release on GitHub. You can follow the progress of the workflow on GitHub; once the workflow completes, there should be a new draft release.

5. Edit the GitHub release. Add release notes (you can use the text generated by towncrier). Check the pre-release checkbox (if necessary).

6. Double check everything, then click Publish. This will trigger a publication workflow on GitHub.

7. Wait for the package to appear on PyPI.

Congratulations, you've just published a release!

If anything went wrong during steps 3 or 5, you will need to delete the draft release from GitHub, and push an updated tag. Once the release has successfully appeared on PyPI, it cannot be changed; if you spot a problem in a published package, you'll need to tag a completely new release.

2.3 About Briefcase

2.3.1 Frequently Asked Questions

What version of Python does toga-chart support?

Python 3.5 or higher.

2.3.2 The toga-chart Developer and User community

Toga-chart is part of the BeeWare suite. You can talk to the community through:

- @pybeeware on Twitter
- BeeWare Getting Help page

Code of Conduct

The BeeWare community has a strict Code of Conduct. All users and developers are expected to adhere to this code.

If you have any concerns about this code of conduct, or you wish to report a violation of this code, please contact the project founder Russell Keith-Magee.

Contributing

If you experience problems with toga-chart, log them on GitHub. If you want to contribute code, please fork the code and submit a pull request.

2.3.3 Release History

0.2.0 (2023-11-03)

Features

- Chart is a now standalone widget, rather than an extension of a Toga Canvas. (#22)

Bugfixes

- The requirements of `toga-chart` were modified so that toga-chart is only dependent on `toga-core`, rather than the `toga` meta-package. This makes it possible to install `toga-chart` on Android, as the meta-package no longer attempts to install the `toga-gtk` backend on Android; but it requires that end-users explicitly specify `toga` or an explicit backend in their own app requirements. (#24)

Misc

- #25, #26, #27, #28, #29, #30, #31, #32, #36, #37, #38, #39, #40, #41, #42, #43, #44, #45, #46, #47, #48, #49, #50, #51, #52, #53, #55, #56, #58, #63

0.1.1 (2022-08-19)

Features

- Chart now has an `on_draw()` handler to ensure the chart auto-resizes. (#14)

Bugfixes

- Added a *None* check before calling *on_draw* (#17)

Misc

- #18

0.1.0 (2020-08-09)

Initial public release.

2.4 Reference

This is the technical reference for public APIs provided by toga-chart.