widgyts

Release 0.3.2

Dec 18, 2019

Table of Contents:

widgyts is a package built on jupyter widgets intended to aid in fast, interactive, exploratory visualization of data. If you have a dataset you'd like to explore but aren't completely sure about what parameters need to be tuned to make the visualization that best representes your data, widgyts is the package for you! widgyts has been designed to work as a companion to yt, but it is also flexible and can handle any mesh-based data.

widyts is a fully client-side pan-and-zoom widget, using WebAssembly, for variable mesh datasets from yt. It runs in the browser, so once the data hits your notebook, it's super fast and responsive! This will allow you to quickly update your visualization to figure out how best to illustrate the interesting aspects of your data.

If you'd like to dig into the Rust and WebAssembly portion of the code, you can find it at Github and in the npm package *@data-exp-lab/yt-tools*.

Check out our SciPy 2018 talk and the associated slides for more info!

CHAPTER 1

Installation

1.1 Dependencies

The widgyts project depends on a number of packages. Minimally, your machine should have `ipywidgets`, `ipydatawidgets` and `yt`.

For a development version, you will also be required to install npm and node.js to manage the javascript code and associated dependencies. Due to the proliferation of the jupyter widgets ecosystem, these are available to install with conda.

1.2 PyPI Installation (recommended)

`widgyts` is packaged and available on the Python Package Index. You can install `widgyts` from pypi by executing:

```
$ pip install widgyts
```

Note that if you do not already have jupyter widgets or jupyter datawidgets installed on your machine or in your current active environment, this step may take some time.

1.3 Installation from Source

To install `widgyts` from source, you'll need to clone the repository from Github:

```
$ git clone https://github.com/data-exp-lab/widgyts.git
```

Then navigate into the newly created directory and install using pip:

```
$ cd widgyts
$ pip install .
```

1.4 Development Installation

For a development installation your machine will need npm to manage the associated javascript code and dependencies.

```
$ git clone https://github.com/data-exp-lab/widgyts.git
$ cd widgyts/js
$ npm install
$ cd ../
$ pip install -e .
$ jupyter serverextension enable --py --sys-prefix widgyts
$ jupyter nbextension install --py --symlink --sys-prefix widgyts
$ jupyter nbextension enable --py --sys-prefix widgyts
```

If you are modifying code on the python side, you may have to periodically update your installation from the steps `pip install` onwards. If you are modifying javascript code, you'll need to rerun `npm install` to have your changes available in jupyter notebooks.

Note that in previous versions, serverextension was not provided and you were required to set up your own mimetype in your local configuration. This is no longer the case and you are now able to use this server extension to set up the correct wasm mimetype.

To install the jupyterlab extension, you will need to make sure you are on a recent enough version of Jupyterlab, preferably 0.35 or above. For a development installation, do:

```
$ jupyter labextension install js
```

To install the latest released version,

```
$ jupyter labextension install @data-exp-lab/yt-widgets
```

CHAPTER 2

Getting Started

The widgyts package is designed to work with yt, but it can also work without a yt import.

2.1 Components of widgyts

Presently widgyts has three widgets that a user can interact with: `ImageCanvas`, `FRBViewer`, and `ColorMaps`. Each widget has a number of traitlets that sync back to the javascript (and potentially webassembly) that can be updated through the widget API. These traitlets can be linked (see our *Examples* for some demonstrations of this in practice) so that widget instances can update together.

2.2 API Documentation

class widgyts.**ImageCanvas**(***kwargs*)
An example widget.

class widgyts.**FRBViewer**(***kwargs*)
View of a fixed resolution buffer.

FRBViewer(width, height, px, py, pdx, pdy, val)

This widget creates a view of a fixed resolution buffer of size (*width*, *height*) given data variables *px*, *py*, *pdx*, *pdy*, and val. Updates on the view of the fixed reolution buffer can be made by modifying traitlets *view_center*, *view_width*, or *Colormaps*

width [integer] The width of the fixed resolution buffer output, in pixels

height [integer] The height of the fixed resolution buffer, in pixels

px [array of floats] x coordinates for the center of each grid box

py [array of floats] y coordinates for the center of each grid box

pdx [array of floats] Values of the half-widths for each grid box

pdy [array of floats] Values of the half-heights for each grid box

val [array of floats] Data values for each grid box The data values to be visualized in the fixed resolution buffer.

colormaps [:class: *widgyts.Colormaps*] This is the widgyt that controls traitlets associated with the colormap.

view_center [tuple] This is a length two tuple that represents the normalized center of the resulting FRBView.

view_width [tuple] This is a length two tuple that represents the height and with of the view, normalized to the original size of the image. (0.5, 0.5) represents a view of half the total data with and half the total data height.

To create a fixed resolution buffer view of a density field with this widget, and then to display it:

```
>>> ds = yt.load("IsolatedGalaxy")
>>> proj = ds.proj("density", "z")
>>> frb1 = widgyts.FRBViewer(height=512, width=512, px=proj["px"],
...                          py=proj["py"], pdx=proj["pdx"],
...                          pdy=proj["pdy"], val = proj["density"])
>>> display(frb1)
```

widgyts.FRBViewer.**setup_controls**(*self*)

widgyts.**display_yt**(*data_object*, *field*)

class widgyts.**ColorMaps**(**args*, ***kwargs*)

Colormapping widget used to collect available colormaps

CHAPTER 3

Examples

The example notebooks illustrate different functionality of the widgyts package. Each is intended to show how widgyts can be used for your own workflow.

- `galaxy_display.ipynb` shows the monkeypatching feature between yt and widgyts to enable you to do interactive visualization directly on any yt dataset.
- `FRBViewer_tutorial.ipynb` shows using the FRBViewer module of widgyts in conjunction with yt's data manipulation features. It illustrates some traitlet manipulation in the Colormaps module as well as traitlet linking between views of different fields.

Links:

- link to galaxy display notebook
- link to FRBViewer tutorial notebook

CHAPTER 4

Contributing

We welcome and encourage new contributors to this project! We're happy to help you work through any issues you're having or to help you contribute to the project, so please reach out if you're interested.

Important: We want your help. No, really.

There may be a little voice inside your head that is telling you that you're not ready to be an open source contributor; that your skills aren't nearly good enough to contribute. What could you possibly offer a project like this one?

We assure you - the little voice in your head is wrong. If you can write code at all, you can contribute code to open source. Contributing to open source projects is a fantastic way to advance one's coding skills. Writing perfect code isn't the measure of a good developer (that would disqualify all of us!); it's trying to create something, making mistakes, and learning from those mistakes. That's how we all improve, and we are happy to help others learn.

Being an open source contributor doesn't just mean writing code, either. You can help out by writing documentation, tests, or even giving feedback about the project (and yes - that includes giving feedback about the contribution process). Some of these contributions may be the most valuable to the project as a whole, because you're coming to the project with fresh eyes, so you can see the errors and assumptions that seasoned contributors have glossed over.

4.1 Issues, Bugs, and New Feature Suggestions

If you have suggestions on new features, improvements to the project itself, you've detected some unruly behavior or a bug, or a suggestion of how we can make the project more accessible, feel free to file an issue on github.

4.2 Communication Channels

If you need help or have any questions about using widgyts that's beyond the documentation (or if you'd like to join our community), you're welcome to join the yt project's slack (specifically in the widgyts channel) or ask in the yt users mailing list.

If you'd like to talk about new features or development of widgyts, the widyts channel in the yt project's slack is a good place to go. The yt development mailing list is channel where these discussions are welcomed.

4.3 Contributing Code

We would be delighted for you to join and work on the widgyts project! If you're interested in getting started, browse some of our open issues and see if there's anything that you may find interesting. If there's a new feature you'd like to add that isn't in open issues, please reach out in the widgyts slack channel in the yt slack or on the yt development mailing list to talk a bit more about what you'd like to contribute. This will help make your contribution review go smoothly and merge quickly!

To get a development environment set up on your machine, please see the *Development Installation* directions to get started.

When issuing a pull request for new features in the widgyts package, please make sure the following are satisfied:

- new features have accompanying documentation, including docstrings and examples
- if javascript functionality is added, ensure it is commented thoroughly
- tests have been added for new features
- all tests run and pass
- new documentation satisfies documentation contribution requirements

4.4 Contributing Examples or Documentation

To contribute new examples or update the documentation you do not need to have a development build of widgyts on your personal machine. To build the documentation, we have opted to use the same method as ipywidgets and distribute an `environment.yml` file that can be used to create an environment with the necessary packages to build the documentation on your personal machine. However, there's no need to create an additional environment if you already have the environment to build the ipywidgets docs.

To install a widgyts docs environment with conda:

```
$ conda env create -f docs/environment.yml
```

and then to activate it:

```
$ source activate widgyts_docs
```

Once the packages necessary to build the documentation are installed on your machine, navigate into the documentation folder and use the Makefile to build the documentation:

```
$ cd docs
$ make clean
$ make html
```

The documentation will be built in the `build/html` folder.

When issuing a pull request for additional documentation or new examples, please make sure the following are satisfied:

- new links, references, and pages work as expected
- the documentation renders locally

- trivializing words like “just”, “simply” or “trivial” are used minimally
- if contributing a notebook, ensure that the data source is clearly documented
- if contributing a notebook, please ensure that each cell has a preamble or comment explaining the contents of the next cell to be executed

4.5 Code Review and Expectations

After you submit a PR with your contribution, you can expect maintainers of the project to begin review within a week.

Please keep in mind that this project is fairly new, so we will try to get back to you as soon as possible with any contributions, but it may take a few days.

Note: We expect members of this community to abide by the *Code of Conduct* when interacting in this community.

CHAPTER 5

Developer Guide

5.1 Getting A Development Build of Widgyts

To get a development build of `widgyts` on your machine, refer to the *Development Installation* instructions.

5.2 Building the Docs

To build the documentation locally, refer to the instructions in the *Contributing Examples or Documentation* section.

5.3 Testing

Our test framework uses pytest. At present, the tests for widgyts are located in the `widgyts/widgyts/tests/` directory. To run the tests, you can execute:

```
$ pytest
```

in the top level widgyts directory. Alternatively, you may choose to run a single file of tests, which can be run using:

```
$ pytest ./widgyts/tests/test_widgyts.py
```

Finally, you can choose to run a single test by calling the specific class and function that run the test within the file by:

```
$ pytest widgyts/tests/test_widgyts.py::TestControls::test_zoom_view
```

5.4 Releasing

CHAPTER 6

Code of Conduct

Note: Widgyts is a project associated with yt, so we have decided to adapt the yt project code of conduct. The yt project code of conduct was adapted from the Astropy Community Code of Conduct, which was partially adapted by the PSF code of conduct.

The community of participants in open source Scientific projects is made up of members from around the globe with a diverse set of skills, personalities, and experiences. It is through these differences that our community experiences success and continued growth. We expect everyone in our community to follow these guidelines when interacting with others both inside and outside of our community. Our goal is to keep ours a positive, inclusive, successful, and growing community.

As members of the community,

- We pledge to treat all people with respect and provide a harassment- and bullying-free environment, regardless of sex, sexual orientation and/or gender identity, disability, physical appearance, body size, race, nationality, ethnicity, and religion. In particular, sexual language and imagery, sexist, racist, or otherwise exclusionary jokes are not appropriate.
- We pledge to respect the work of others by recognizing acknowledgment/citation requests of original authors. As authors, we pledge to be explicit about how we want our own work to be cited or acknowledged.
- We pledge to welcome those interested in joining the community, and realize that including people with a variety of opinions and backgrounds will only serve to enrich our community. In particular, discussions relating to pros/cons of various technologies, programming languages, and so on are welcome, but these should be done with respect, taking proactive measure to ensure that all participants are heard and feel confident that they can freely express their opinions.
- We pledge to welcome questions and answer them respectfully, paying particular attention to those new to the community. We pledge to provide respectful criticisms and feedback in forums, especially in discussion threads resulting from code contributions.
- We pledge to be conscientious of the perceptions of the wider community and to respond to criticism respectfully. We will strive to model behaviors that encourage productive debate and disagreement, both within our community and where we are criticized. We will treat those outside our community with the same respect as people within our community.

- We pledge to help the entire community follow the code of conduct, and to not remain silent when we see violations of the code of conduct. We will take action when members of our community violate this code such as contacting confidential@yt-project.org (all emails sent to this address will be treated with the strictest confidence) or talking privately with the person.

This code of conduct applies to all community situations online and offline, including mailing lists, forums, social media, conferences, meetings, associated social events, and one-to-one interactions.

CHAPTER 7

Getting Help

Do not hesitate to reach out through one of our numerous communication channels should you need help with anything widgyts-related.

7.1 Communication Channels

If you need help or have any questions about using widgyts that's beyond the documentation (or if you'd like to join our community), you're welcome to join the yt project's slack (specifically in the widgyts channel) or ask in the yt users mailing list.

If you'd like to talk about new features or development of widgyts, or if you need some assistance developing in the widgyts project, the widyts channel in the yt project's slack is a good place to go. The yt development mailing list is channel where these discussions are also welcomed.

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Index