STScI | SPACE TELESCOPE SCIENCE INSTITUTE

EXPANDING THE FRONTIERS OF SPACE ASTRONOMY

Implementing MIVOT into PyVO

Tom Donaldson

IVOA Interop – Northern Fall 2022

Astropy and PyVO

Astropy – A Community Python Library for Astronomy

- The `astropy` package contains key functionality and common tools needed for performing astronomy and astrophysics with Python. It is at the core of the Astropy Project, which aims to enable the community to develop a robust ecosystem of affiliated packages covering a broad range of needs for astronomical research, data processing, and data analysis.
- Open development process
  - All are welcome and encouraged to contribute
  - See the Developer Documentation for all the details
- GitHub: https://github.com/astropy/astropy

PyVO – A Community Python Library for Accessing the Virtual Observatory

- One of Astropy's affiliated packages
- Uses the same open development process as Astropy
- GitHub: https://github.com/astropy/pyvo
- Astropy Slack has a #pyvo channel (get an account)

Astropy VOTable Parser

PyVO relies on Astropy's VOTable Parser

- Submodule `astropy.io.votable`
- Astropy changes for MIVOT (if any) would likely be there
  - e.g., to ensure that MIVOT artifacts are retained and available after parsing
- ⚠️ Note that astropy will soon be forcing the use of Black for code formatting
  - See APE-20 for how this will work

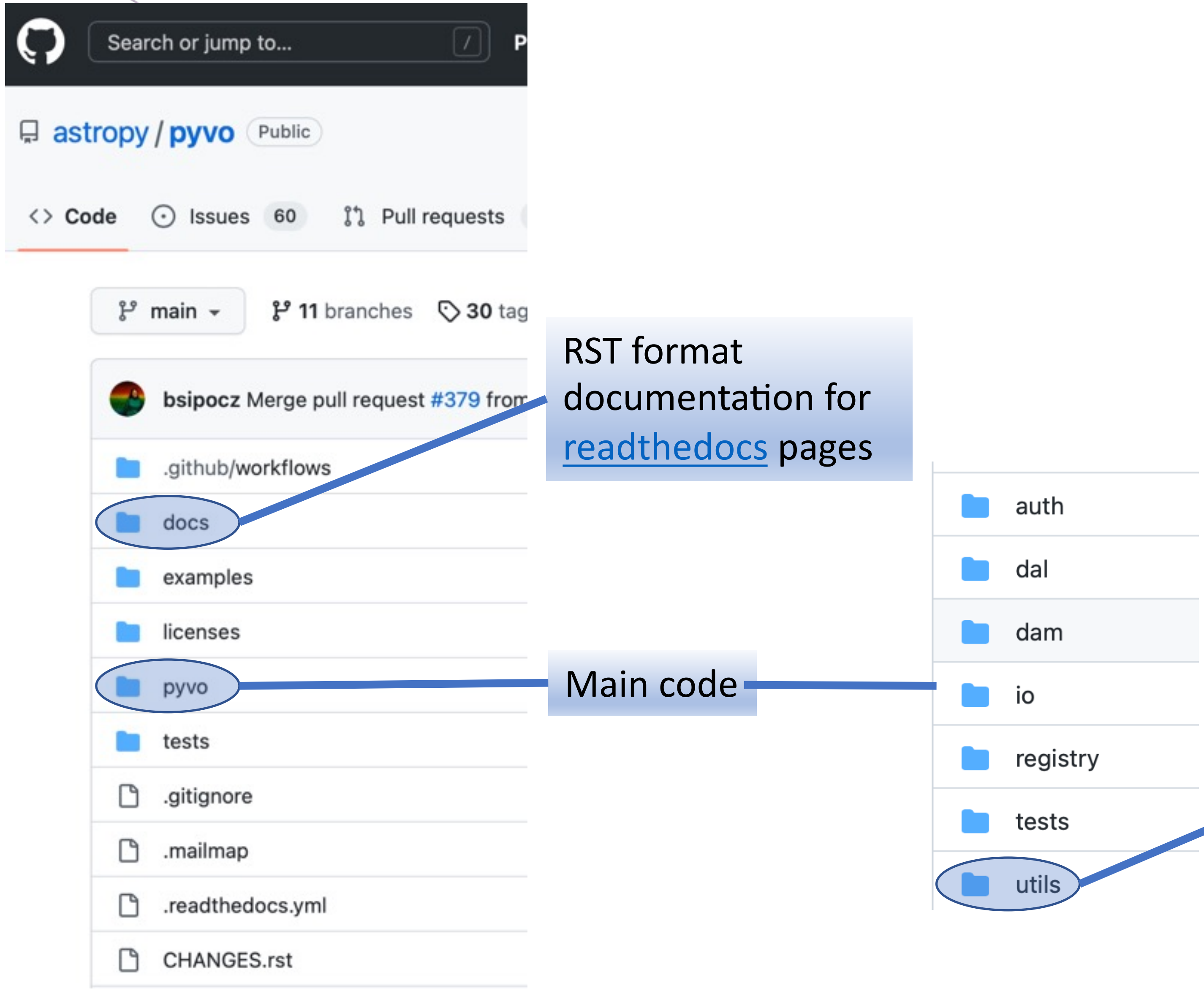
See Astropy VOTable Documentation for API details

- To read in a VOTable file, pass a file path to parse:

```
from astropy.io.votable import parse
votable = parse("votable.xml")
```

- votable is a VOTableFile object, which can be used to retrieve and manipulate the data and save it back out to disk, or to an Astropy table using `to_table()`
- Note this is integrated with Astropy table read() and write() using format='votable'

PyVO Components


Search or jump to...

astropy / pyvo Public

Code   Issues 60   Pull requests

main   11 branches   30 tag

bsipocz Merge pull request #379 from

.github/workflows
docs
examples
licenses
pyvo
tests
.gitignore
.mailmap
.readthedocs.yml
CHANGES.rst

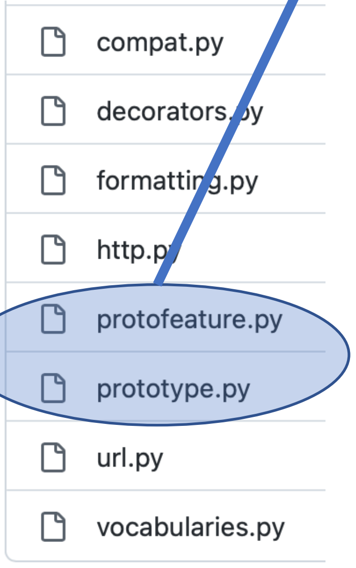
auth
dal
dam
io
registry
tests
utils

RST format documentation for readthedocs pages

Main code

Prototype feature

- made by Omar Laurino
- Encapsulates code for standards in progress

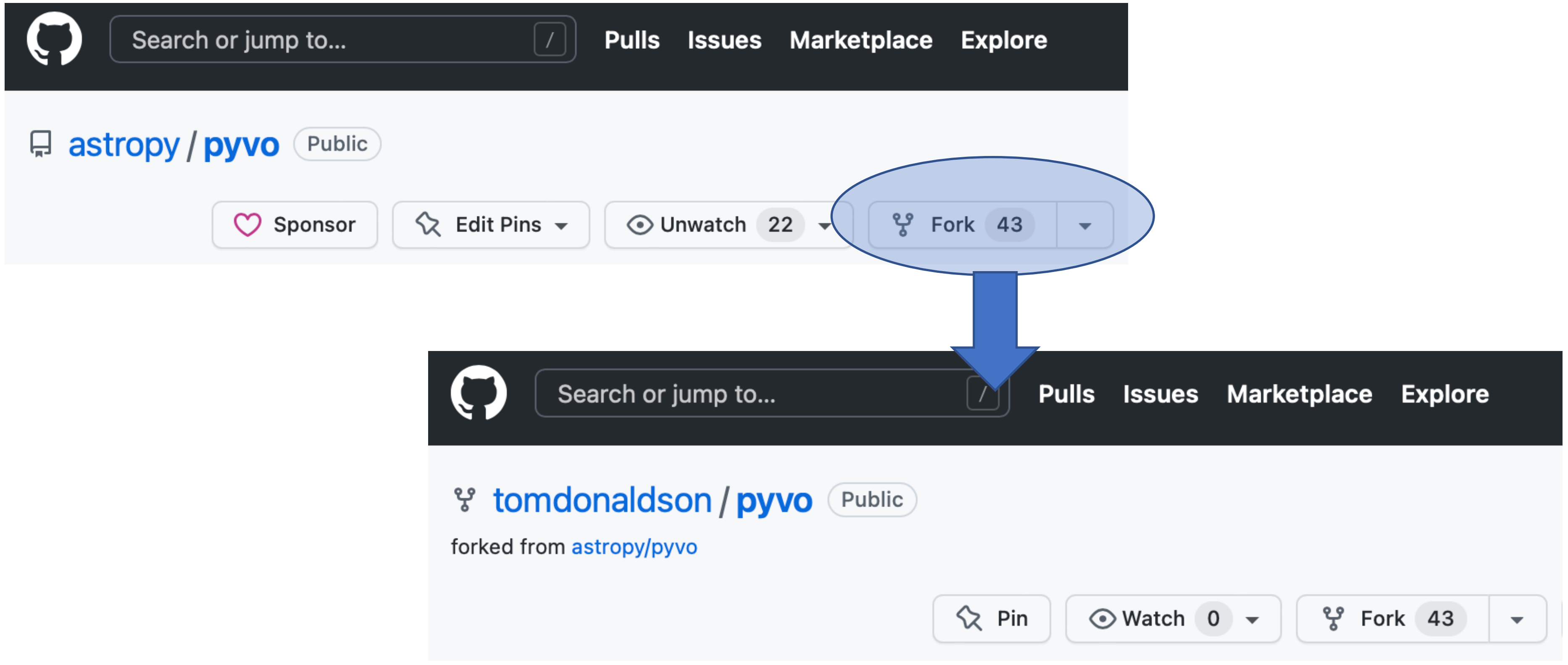

compat.py
decorators.py
formatting.py
http.py
protofeature.py
prototype.py
url.py
vocabularies.py

Development Process (same for Astropy and PyVO)

See the Astropy Developer Documentation for all the details! Summary below…

Fork the repository (one time only)

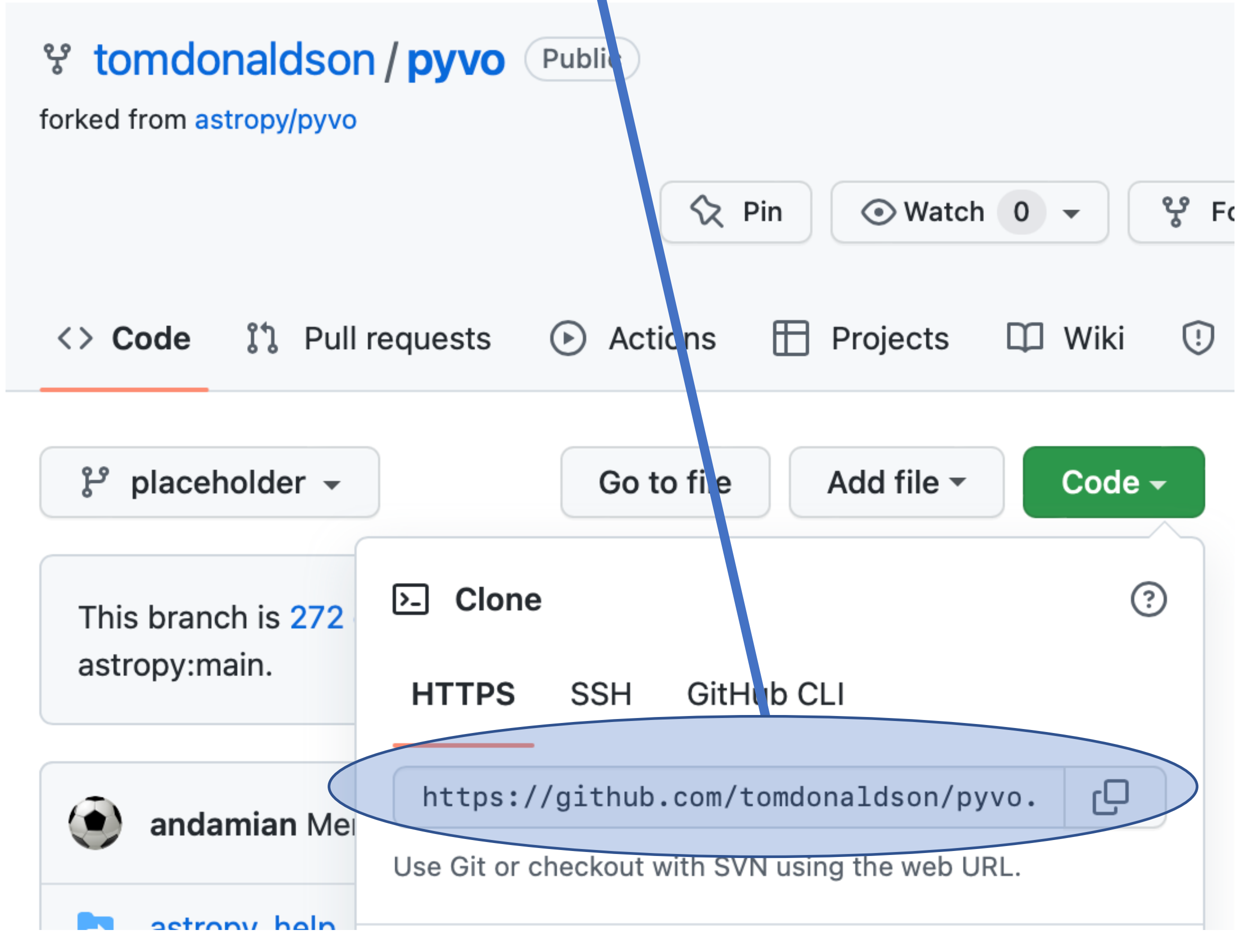
- (Optional) delete your main branch



Development Process - Clone

Clone your own repository (one or more times)

```
git clone https://github.com/<yourgithubname>/pyvo.git
cd pyvo
```



```
https://github.com/tomdonaldson/pyvo.
```

Use Git or checkout with SVN using the web URL.

Development Process – Create Branch

(Optional) Add easy reference main astropy/pyvo repository

```
git remote add pyvo https://github.com/astropy/pyvo.git
```

Create (and checkout!) branch based on pyvo/main

```
# Make sure local repo has everything from upstream
git fetch pyvo –tags

# Create and checkout your branch
git branch mivot-work pyvo/main
git checkout mivot-work

# (Optional) Have commits automatically go to your new branch on your github
git push --set-upstream origin mivot-work
```

Development Process – Create Development Environment

Conda or Python Virtual Environment

```
conda create --name mivot-work-env python=3.8
conda activate mivot-work-env

# Install pyvo from the local clone; code changes are part of environment
pip install –e .

# Install other packages that may be useful during development
pip install -U tox pytest-astropy requests_mock pillow
```

Development Process – Local Testing

Run tests locally

- Worth checking that everything is working before you make changes

```
tox -e test          # local only
pytest               # Alternative to tox that does less setup

tox -e test-online   # with remote-data
```

- Can also check codestyle (more useful after you've made changes)

```
tox -e codestyle
```

Then make your code changes

- Don't forget to add tests and documentation changes!

Development Process – Commits and Pull Requests

Commit and push changes (as often as needed)

```
git commit -m "My awesome changes"    # Saves changes in local repo
git push                              # Saves commits to your branch on github
```

Create Pull Request (in GitHub page for your repo)

- Ensure that tox test and codestyle work locally
- Ensure all local code is committed and pushed
- Use Draft PRs to stimulate early discussion and reviews, esp. for larger projects.
- Does not all need to be in a single PR
- Request reviewers
  - Need not be PyVO maintainers, they will see the PR anyway
- Respond to comments on the PR until it's approved and merged

Use PyVO GitHub issues to report problems and start discussions