OSS Module 02 - Git Workflows - Issues, Forks, and Pull Requests

Contents

- 1: About this training module
- 2: Issues
- 3: Pull requests
- 4: Forks
- 5: The fork/pull workflow

1: About this training module

Getting Started

This training module will introduce you to collaborative workflows on GitHub. We will look at some real world workflows from the NumPy repo. Then we will proceed with some exercies in the `gt-ospo` GitHub organization.

It is highly recommended that you read Module 1: Git before proceeding with this module.

The excercies at the end require you to:

1. Have Git on your computer
2. Have a GitHub account
3. Be a member of the gt-ospo Git organization

One size doesn't fit all!

GitHub offers a wealth of collaborative tools. We are covering the most common ones today, which are well-documented on GitHub. We expect most projects will make use of these tools. However, many project's will use more, and some will use fewer, depending on the project. Please defer to your project's maintainers about their workflows.

1: Issues

What is an Issue?

Before we talk about actually contributing new code, let's talk about beginning the conversation with **Issues**.

When you create an issue, you are starting a conversation thread about a specific topic with the project's maintainers, uses, and other stakeholders. Many projects will use issues for suggesting feature enhancements and reporting bugs.

Issues have a huge number of benefits. Here are just a few:

- When you want to contribute features or bugfixes to an existing project, you first want to make sure the changes are in the interests of the stakeholders. You don't waste your time or the maintainers' time by developing code that is not of interest. Starting with Issue can ensure that you and the stakeholders are on the same page before you start work.
- They can be linked to Pull Requests, where new code is actually contributed. More on that below...
- It serves as a record of project development and bug reports. Many times, when you encounter a bug in a project, you can search historical issues on GitHub to find an answer to your problem.
- They are an excellent fit for project management. Issues can tranlate to tasks and be tracked by project management tools, like a Kanban board. Hence, even internal projects will often use Issues to track the internal team's work.

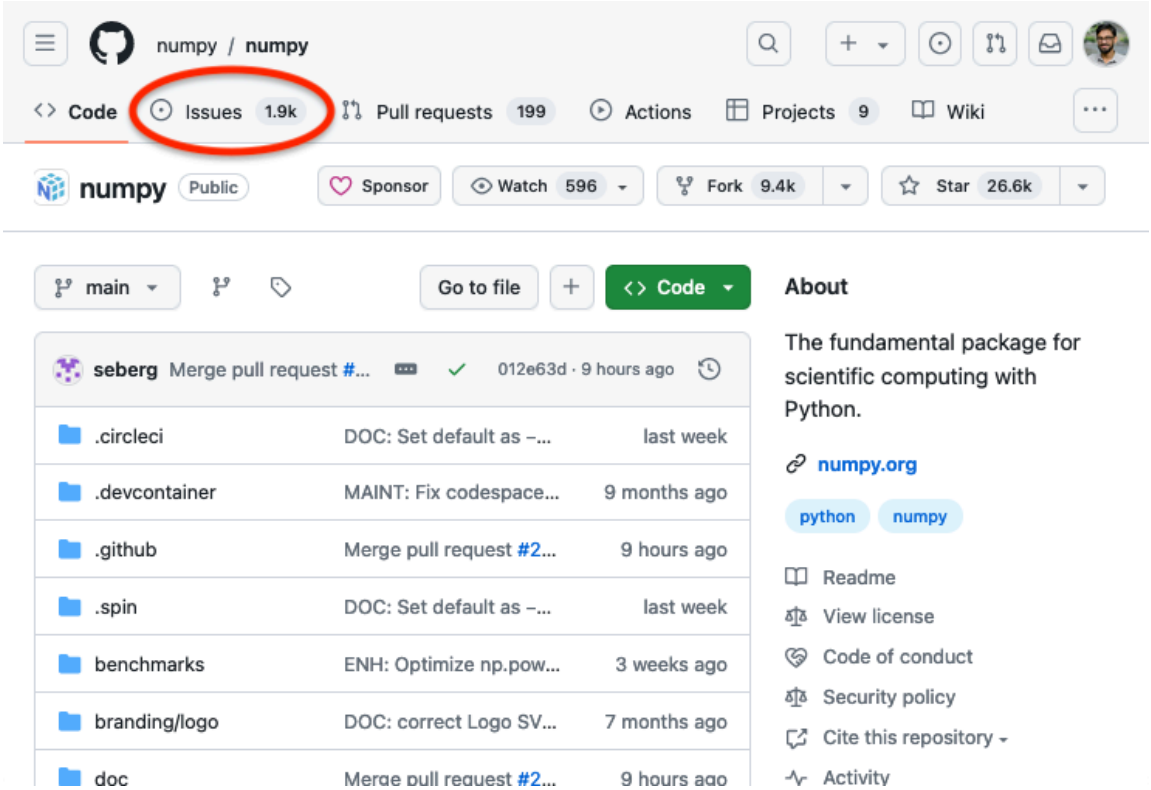
The GitHub docs have much more info in: "Tracking your work with Issues"

About codes of conduct

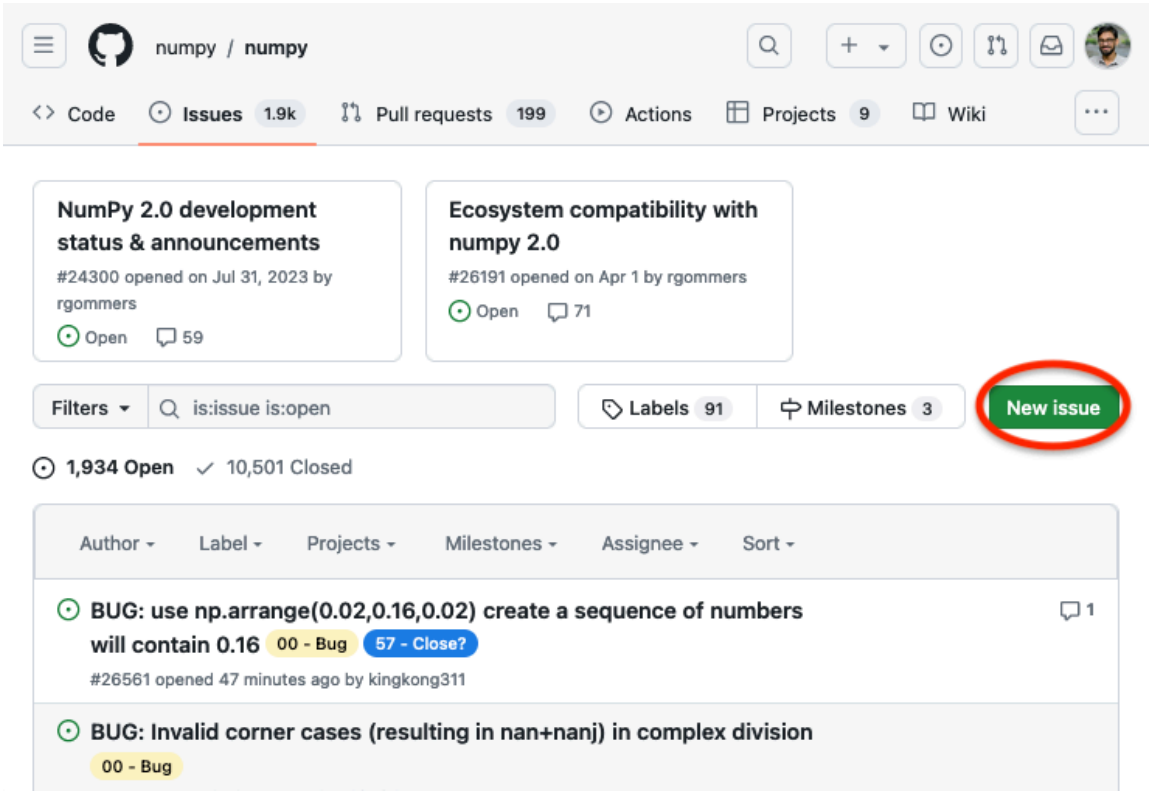
Issues are one of many ways to have conversations on GitHub. As with any online conversations, mutual respect is crucial. Most projects (such as ours!) have a code of conduct to help keep conversations respectful. This makes sure everyone has productive (and hopefully fun!) experience.

A real-world example: Issues in NumPy

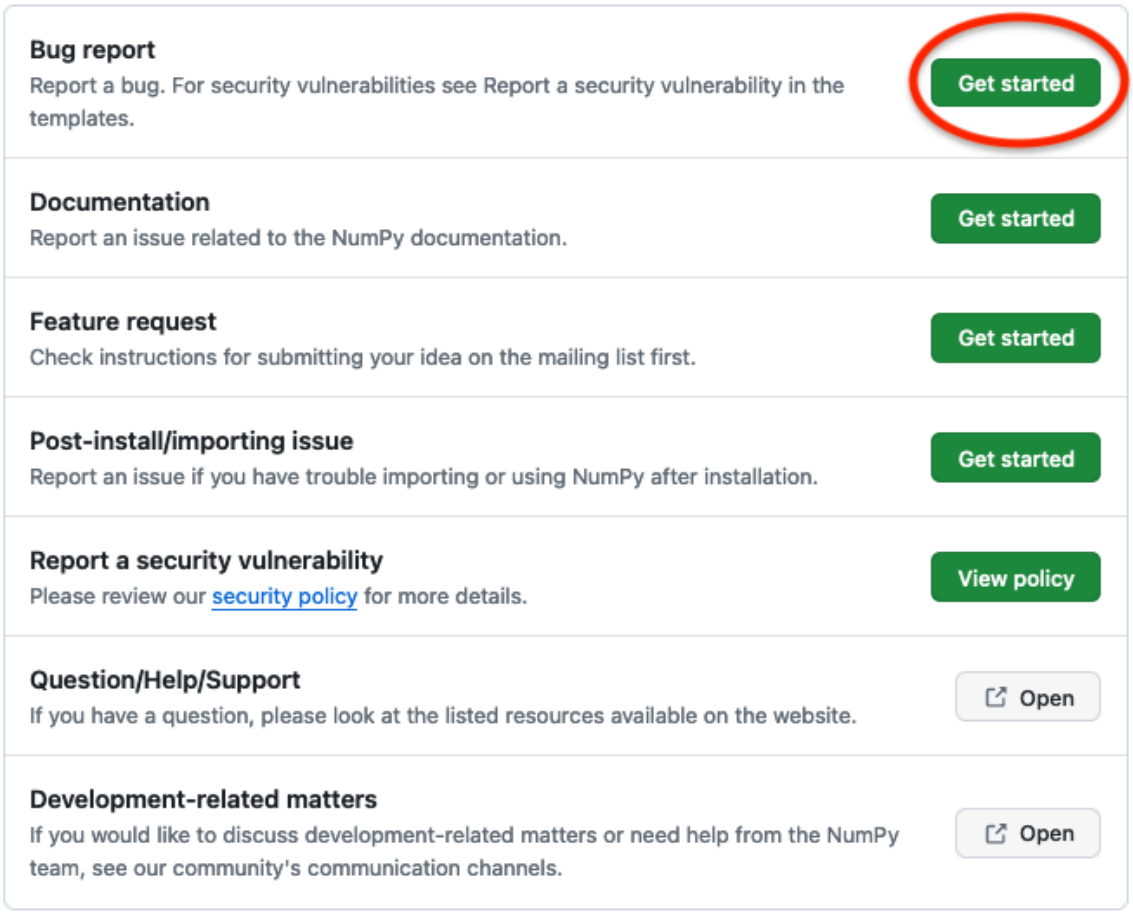
NumPy is a huge project. To manage it, the team uses very well-structured issues. Let's begin by looking at the issues page:



On the issues page, you can create a new issue:

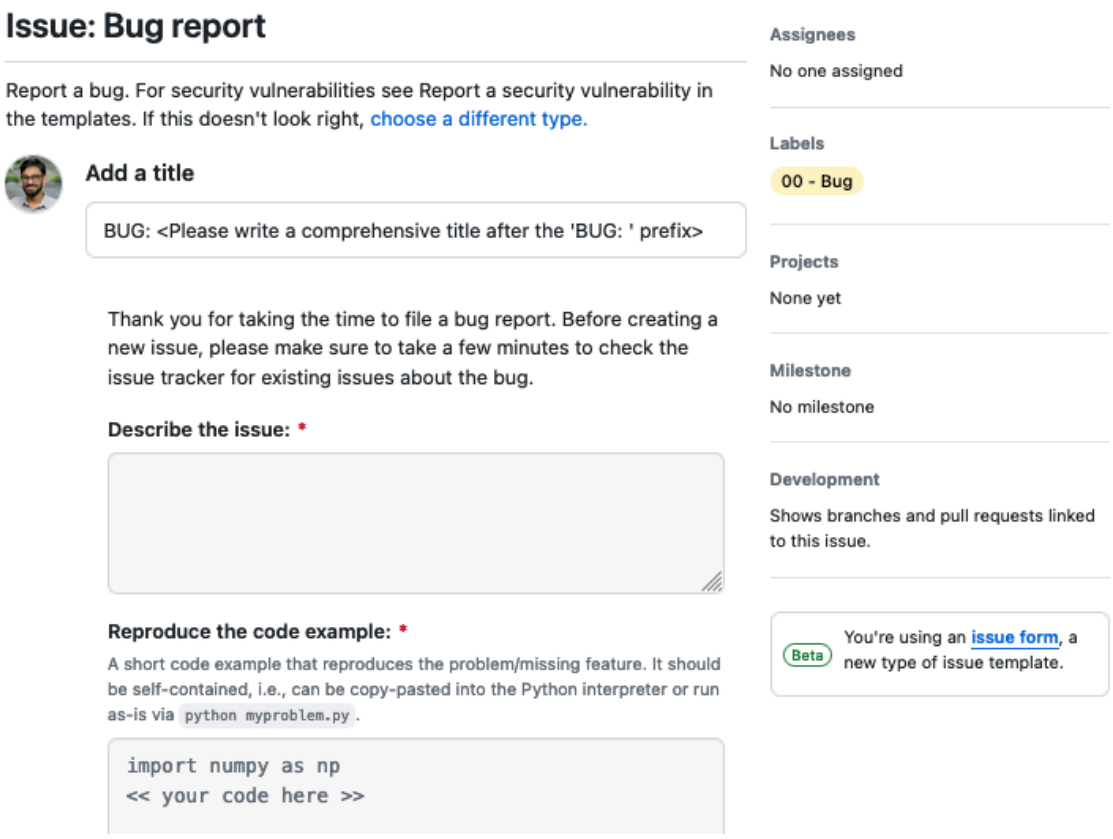

numpy / numpy
Code
Issues 1.9k
Pull requests 199
Actions
Projects 9
Wiki
NumPy 2.0 development status & announcements
#24300 opened on Jul 31, 2023 by rgommers
Open 59
Ecosystem compatibility with numpy 2.0
#26191 opened on Apr 1 by rgommers
Open 71
Filters
is:issue is:open
Labels 91
Milestones 3
New issue
1,934 Open
10,501 Closed
Author
Label
Projects
Milestones
Assignee
Sort
BUG: use np.arrange(0.02,0.16,0.02) create a sequence of numbers will contain 0.16
00 - Bug
57 - Close?
1
#26561 opened 47 minutes ago by kingkong311
BUG: Invalid corner cases (resulting in nan+nanj) in complex division
00 - Bug

When you create a issue, you are first prompted to chose they type of issue: A bug report, a feature request, a documentation issue, and others:

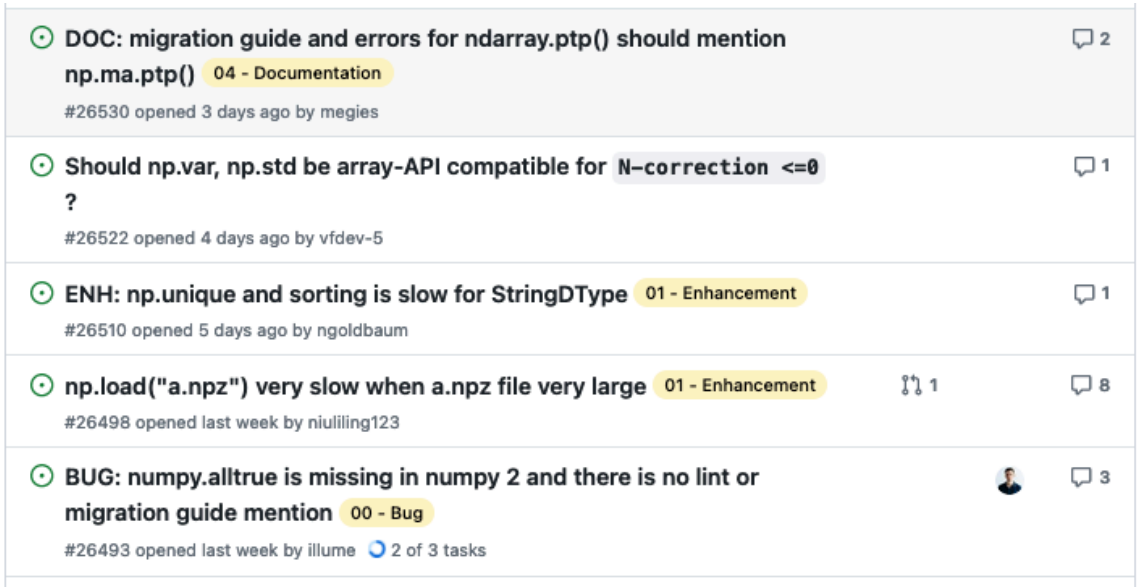


Let's look at the form for a bug report.

- NumPy has a detailed form with required info. This helps the NumPy devs understand your issue and deal with it efficiently.
- The issue is auto-tagged with a **label** named "00 - Bug"



Labels are ways to help humans organize issues. Going back to main issues page, can see issues with other labels. Labels for bugs, enchancements, and documentation are used by many projects. Many projects all labels for a good first issue for new contributors (for example, pandas).



Issues in smaller projects

Because NumPy is such a huge project, it makes use of many high-level GitHub features for managing issues. Many projects will not make use every single feature. For example, NumPy uses the newer Issue Forms feature, but many repos use the simpler Issue Templates feature

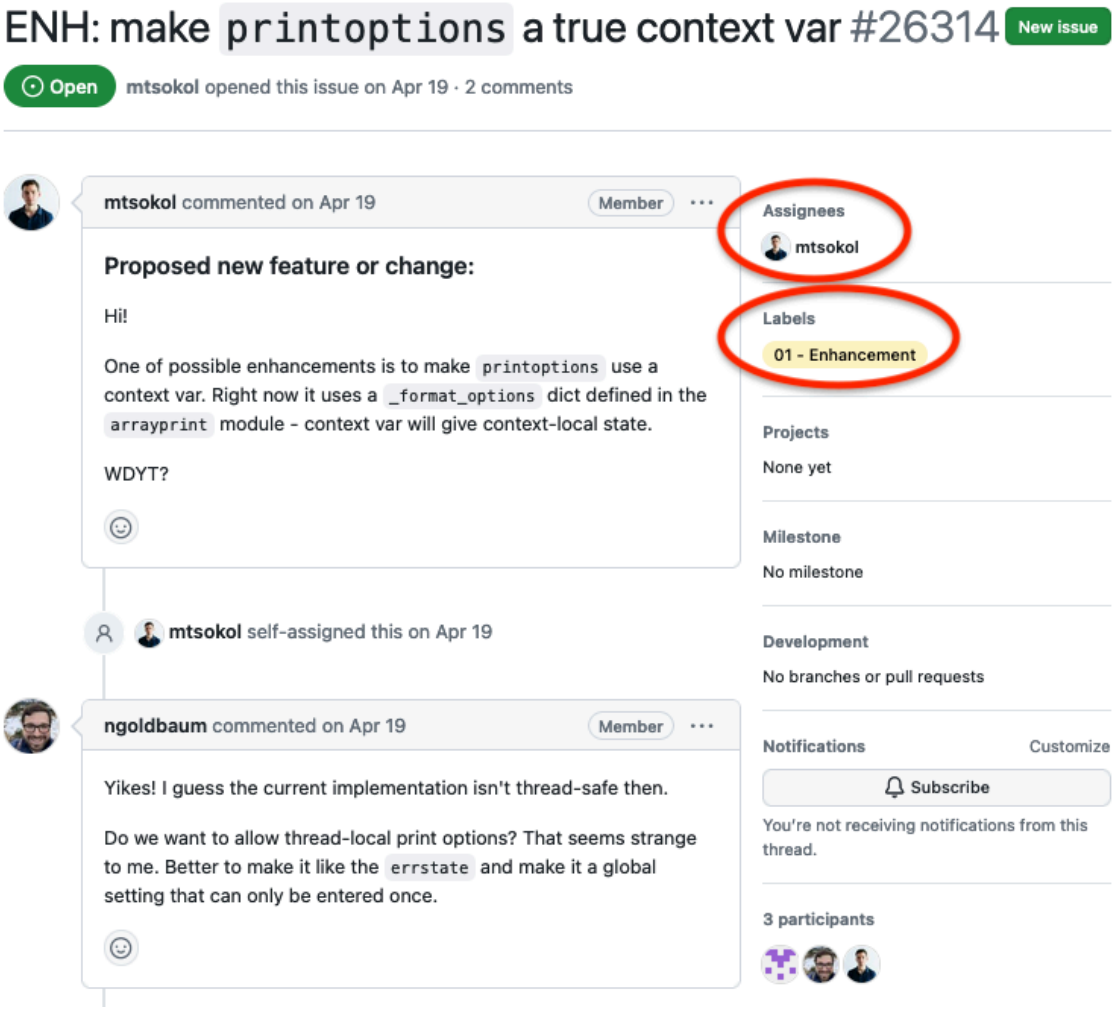
Some features that are used by both large and small projects:

- Labels
- Assignees
- Linking to a pull request (more below...)

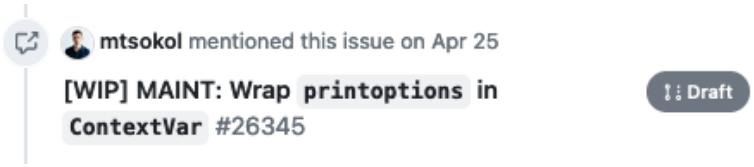
How issues get discussed and resolved

Let's look at an existing Issue in NumPy, #26314, to see how issues are discussed and resolved.

- The initial post has a proposal for a new feature.
- It was tagged with "01 - Enhancement"
- There was a follow-up conversation with other devs. This made it clear that the feature was desirable.
- The issue was assigned to a developer for follow-up. In this case, the assignee was the original submitter. In other cases (for example, bug reports from users), the assignee might be a different person from the submitter.



Finally, note that this issue was linked to the **pull request** that addresses the issue. We will discuss pull requests next.



Pull requests

- How big should a PR be (lines of code, scope, ...)
- What should you write in the PR description?
- How are PRs reviewed?
- Refs:
  - https://opensource.com/article/18/6/anatomy-perfect-pull-request
  - https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/getting-started/best-practices-for-pull-requests
  - https://github.blog/2015-01-21-how-to-write-the-perfect-pull-request/

Pull requests

As described by the GitHub docs "About pull requests"

"A **pull request** is a proposal to merge a set of changes from one branch into another. In a pull request, collaborators can review and discuss the proposed set of changes before they integrate the changes into the main codebase."

It can be helpful to think of a pull requests as a way to do collaborative `git merge` on a remote repo, rather than in one person's local working copy of a repo. And indeed, on the backend of GitHub, a pull request is ultimately ful

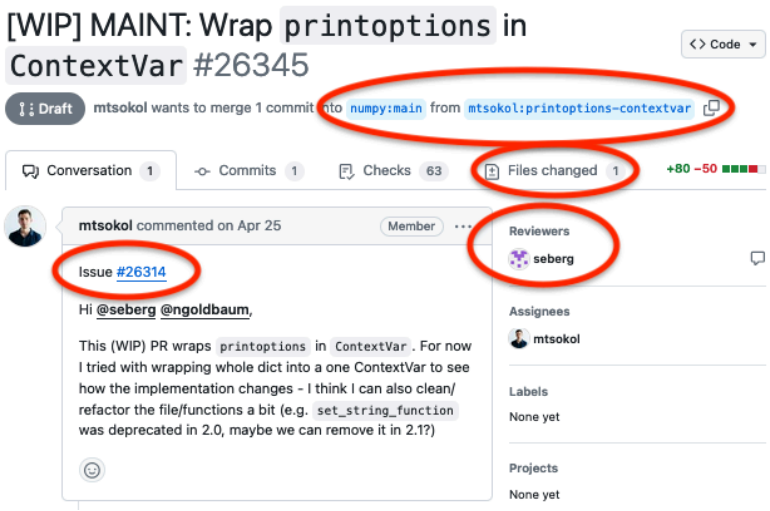
The Pull Request is a **proposal** that is intended to initiate a conversation with the project maintainers. The merge does not automatically happen *until* a human completes it. Submitting a PR can additionally trigger automated testing. The repo is often configured with controls to enforce a specific process:

- Require a review from designated reviewer(s) before merging
- Require all tests to pass before merging

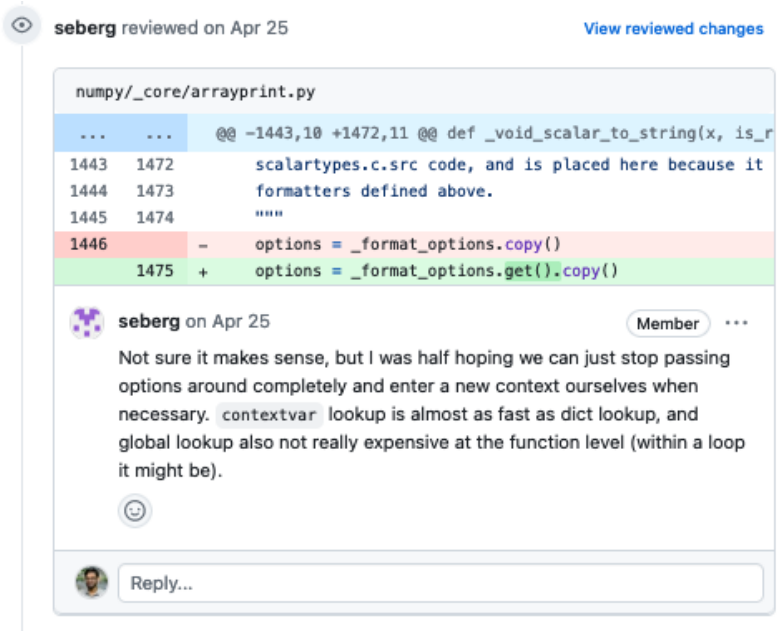
Returning to Issue #26314 in NumPy, let's look at the pull request that was linked to it: PR #26345.

We see that:

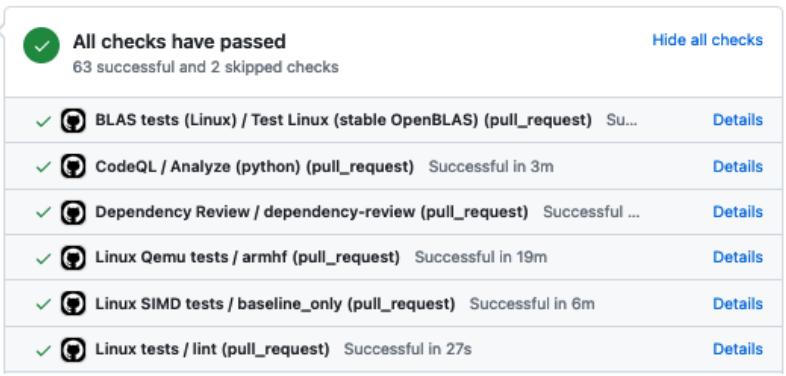
- The submitter is requesting to merge the topic branch `mtsokol:printoptions-contextvar` into the **target branch** `numpy:main`
- The submitter has linked it to an Issue by writing "Issue #26314" in the description. Linking a PR to an issue is not required, but it can be extremely helpful for managing your project. See the GitHub docs, "Linking a pull request it an issue for more details
- Another developer has been assigned to review it
- You can click on "Files changed" to see the diff between the topic branch and the target branch



Reviews can be very detailed with line-by-line conversation threads:



This repo has automated tests. This repo has code quality tests (like a linter), static analyis tools (mypy) and tests for the actual functionality. We'll learn more about automated testing in Module 03



Looking back at the top of the PR, we saw that the submitter is requesting to merge the topic branch `mtsokol:printoptions-contextvar` into the target branch `numpy:main`. This means that:

- The topic branch was on the submitters **copy** of the repo, https://github.com/mtsokol/numpy
- The target branch was on the original repo, https://github.com/numpy/numpy

The submitter's copy is called a **fork** or the original. We'll talk about that next.

Forks

By creating a **fork**, you are creating a copy of another repo that:

- Shares the history of the original
- Is linked to the original for the purposes of pull requests.

In a forking workflow, the original copy is called the **upstream**.

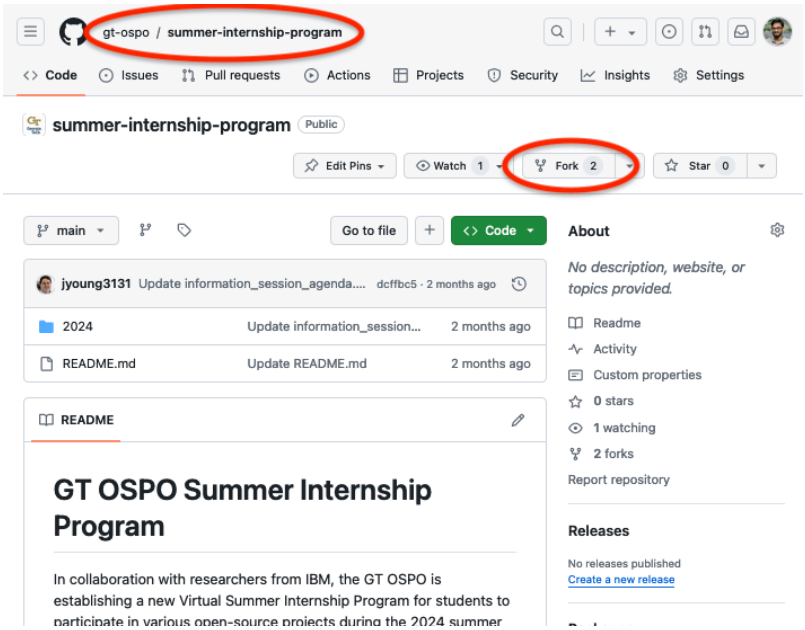
In your fork, you can make changes to a project without affecting the upstream. Then when you're ready to propose an update to upstream, you submit a pull request. When you're commiting, pushing, pulling, and doing other Git tasks with your fork, **it acts exactly like any other repo**.

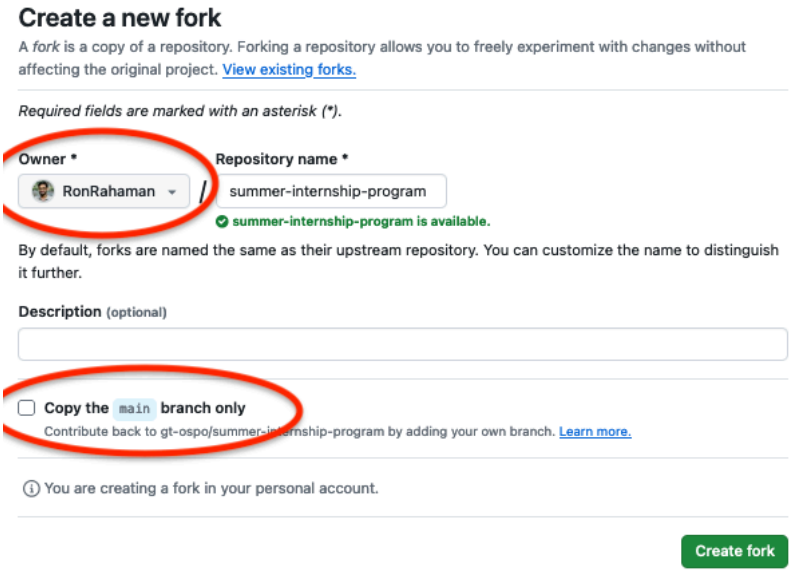
There is much more info on the GitHub docs, "About forks"

Creating a fork

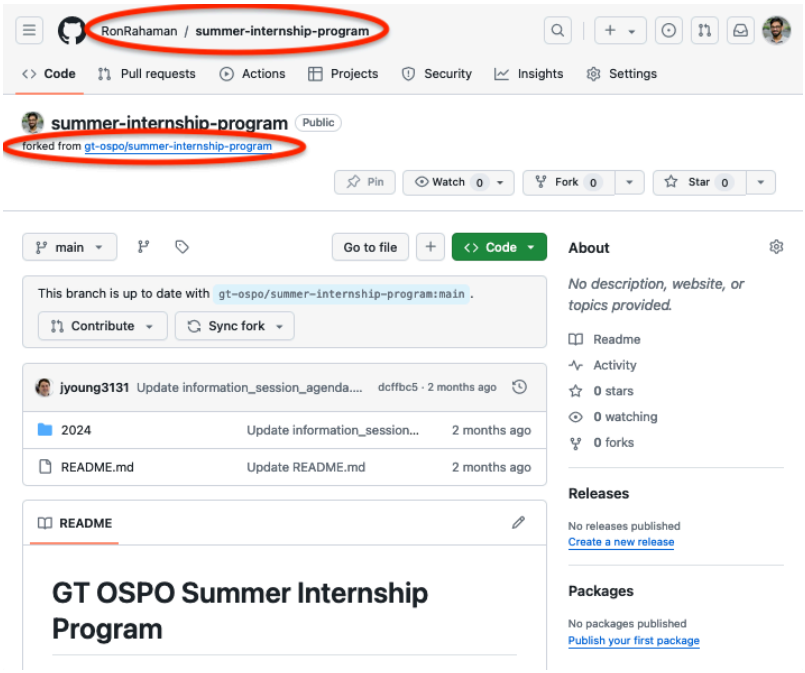
Start from: https://github.com/gt-ospo/summer-internship-program

- Owned by the organization `gt-ospo`





- Creating this in my personal account (could be another org
- Getting all the branches (I like to do this, but might not want to do that if there are a whole bunch of branches)



Now I have my own repo at: https://github.com/RonRahaman/summer-internship-program Share a history with the upstream

Working with your fork

At this point, I can clone my fork as usual:

In [6]:
```
git clone git@github.com:RonRahaman/summer-internship-program.git
```

```
Cloning into 'summer-internship-program'...
remote: Enumerating objects: 88, done.
remote: Counting objects: 100% (88/88), done.
remote: Compressing objects: 100% (83/83), done.
remote: Total 88 (delta 19), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (88/88), 19.74 MiB | 8.82 MiB/s, done.
Resolving deltas: 100% (19/19), done.
```

In [7]:
```
cd summer-internship-program
```

In my working copy, I my fork is named "origin"

In [8]:
```
git remote -v
```

```
origin  git@github.com:RonRahaman/summer-internship-program.git (fetch)
origin  git@github.com:RonRahaman/summer-internship-program.git (push)
```

Now, I will add the original repo as an upstream.

Recall that we can use the command `git remote add NAME URL` to add a new remote. The NAME can be anything and doesn't affect GitHub. But to keep myself sane, I'll follow the widespread convention that the original repo is named "upstream"

In [9]:
```
git remote add upstream git@github.com:gt-ospo/summer-internship-program.git
```

Now we can see both remotes:

In [10]:
```
git remote -v
```

```
origin  git@github.com:RonRahaman/summer-internship-program.git (fetch)
origin  git@github.com:RonRahaman/summer-internship-program.git (push)
upstream        git@github.com:gt-ospo/summer-internship-program.git (fetch)
upstream        git@github.com:gt-ospo/summer-internship-program.git (push)
```

But at this point, you will only see remote tracking branches from "origin". Recall that `origin/main` shows the known status of the branch "main" on the remote "origin"

In [11]:
```
git log --remotes --oneline
```

```
dcffbc5 (HEAD -> main, origin/main, origin/HEAD) Update information_session_
agenda.md
841cca0 Update information_session_agenda.md
c30f9c3 Added VSIP introduction slides
cbf5fcc Delete 2024/project_slides/.gitkeep
aad7b66 Merge pull request #1 from sbryngelson/main
1b64ea5 Update information_session_agenda.md
adb2b7f Update information_session_agenda.md
b0f9827 FL: Updated three slide links
0e769d2 FL: added two more presentation
8a7ddf8 FL: Updte trestle
992f802 Update information_session_agenda.md
b9c8deb Update README.md
a9ba7b5 Update information_session_agenda.md
a2c9156 Update agenda for Summer 2024 session
a7b1b45 Update information_session_agenda.md
7fe5e27 FL: Upload 12 IBM projects with naming conversion
304bf98 Create information_session_agenda.md
f018ed8 Create .gitkeep
682ca09 Initial commit
```

To get this remote tracking info, we use `git fetch`. Recall that `git fetch` will update our remote tracking branches. We could use `git fetch upstream` to fetch remote tracking branches from upstream. We can also use `git fetch --all` if we want to fetch from all our remotes at once.

In [12]:
```
git fetch upstream
```

```
From github.com:gt-ospo/summer-internship-program
 * [new branch]      main       -> upstream/main
```

Now we can see that we have remote tracking branches for upstream, too. Note that `origin/main`, `upstream/main`, and `main` are all in the same place now.

In [13]:
```
git log --remotes --oneline
```

```
dcffbc5 (HEAD -> main, upstream/main, origin/main, origin/HEAD) Update infor
mation_session_agenda.md
841cca0 Update information_session_agenda.md
c30f9c3 Added VSIP introduction slides
cbf5fcc Delete 2024/project_slides/.gitkeep
aad7b66 Merge pull request #1 from sbryngelson/main
1b64ea5 Update information_session_agenda.md
adb2b7f Update information_session_agenda.md
b0f9827 FL: Updated three slide links
0e769d2 FL: added two more presentation
8a7ddf8 FL: Updte trestle
992f802 Update information_session_agenda.md
b9c8deb Update README.md
a9ba7b5 Update information_session_agenda.md
a2c9156 Update agenda for Summer 2024 session
a7b1b45 Update information_session_agenda.md
7fe5e27 FL: Upload 12 IBM projects with naming conversion
304bf98 Create information_session_agenda.md
f018ed8 Create .gitkeep
682ca09 Initial commit
```

I'm going to make a change to `README.md`, then commit and push it to my fork.

In [14]:
```
git checkout -b module02-training
```

```
Switched to a new branch 'module02-training'
```

I corrected a typo in the text editor. Here is the diff:

In [15]:
```
git diff
```

```
diff --git a/README.md b/README.md
index 699d321..cccbc96 100644
--- a/README.md
+++ b/README.md
@@ -1,5 +1,5 @@
 # GT OSPO Summer Internship Program

-In collaboration with researchers from IBM, the GT OSPO is establishing a n
ew Virtual Summer Internship Program for students to participate in various
open-source projects during the 2024 summer session. During the internship,
students will join an open source project, work with the developers and main
tainers of that project to make contributions to the code base, and learn us
eful software development and engineering skills to engage with the wider op
en source community.
+In collaboration with researchers from IBM, the GT OSPO is establishing a n
ew Virtual Summer Internship Program for students to participate in various
open-source projects during the 2024 summer session. During the internship,
students will join an open source project, work with the developers and main
tainers of that project to make contributions to the code base, and learn us
eful software development and engineering skills to engage with the wider op
en-source community.

 If you are interested to learn more about this internship, please see the
[main VSIP program page here](https://ospo.cc.gatech.edu/vsip/) and the [age
nda for the February 27th event](https://github.com/gt-ospo/summer-internshi
p-program/blob/main/2024/information_session_agenda.md).
```

In [16]:
```
git status
```

```
On branch module02-training
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .ipynb_checkpoints/

no changes added to commit (use "git add" and/or "git commit -a")
```

In [17]:
```
git add README.md
```

In [18]:
```
git commit -m "RR: Correct hyphen in 'open-source'"
```

```
[module02-training a2d0a7b] RR: Correct hyphen in 'open-source'
 1 file changed, 1 insertion(+), 1 deletion(-)
```

In [20]:
```
git log --remotes --oneline --branches
```

```
a2d0a7b (HEAD -> module02-training) RR: Correct hyphen in 'open-source'
dcffbc5 (upstream/main, origin/main, origin/HEAD, main) Update information_s
ession_agenda.md
841cca0 Update information_session_agenda.md
c30f9c3 Added VSIP introduction slides
cbf5fcc Delete 2024/project_slides/.gitkeep
aad7b66 Merge pull request #1 from sbryngelson/main
1b64ea5 Update information_session_agenda.md
adb2b7f Update information_session_agenda.md
b0f9827 FL: Updated three slide links
0e769d2 FL: added two more presentation
8a7ddf8 FL: Updte trestle
992f802 Update information_session_agenda.md
b9c8deb Update README.md
a9ba7b5 Update information_session_agenda.md
a2c9156 Update agenda for Summer 2024 session
a7b1b45 Update information_session_agenda.md
7fe5e27 FL: Upload 12 IBM projects with naming conversion
304bf98 Create information_session_agenda.md
f018ed8 Create .gitkeep
682ca09 Initial commit
```

In [21]:
```
git push -u origin module02-training
```

```
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 324 bytes | 324.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'module02-training' on GitHub by visiting:
remote:      https://github.com/RonRahaman/summer-internship-program/pull/ne
w/module02-training
remote:
To github.com:RonRahaman/summer-internship-program.git
 * [new branch]      module02-training -> module02-training
Branch 'module02-training' set up to track remote branch 'module02-training'
from 'origin'.
```

Submitting a pull request

Now I will begin the process of submitting a pull request.

Updating your fork from upstream

After it's complete, let's go back to our working copy and check on our branches. As we'd expect, the remote tracking branches are *not* yet up-to-date since we haven't explicitly fetched them

In [22]:
```
git log --remotes --oneline
```

```
a2d0a7b (HEAD -> module02-training, origin/module02-training) RR: Correct hy
phen in 'open-source'
dcffbc5 (upstream/main, origin/main, origin/HEAD, main) Update information_s
ession_agenda.md
841cca0 Update information_session_agenda.md
c30f9c3 Added VSIP introduction slides
cbf5fcc Delete 2024/project_slides/.gitkeep
aad7b66 Merge pull request #1 from sbryngelson/main
1b64ea5 Update information_session_agenda.md
adb2b7f Update information_session_agenda.md
b0f9827 FL: Updated three slide links
0e769d2 FL: added two more presentation
8a7ddf8 FL: Updte trestle
992f802 Update information_session_agenda.md
b9c8deb Update README.md
a9ba7b5 Update information_session_agenda.md
a2c9156 Update agenda for Summer 2024 session
a7b1b45 Update information_session_agenda.md
7fe5e27 FL: Upload 12 IBM projects with naming conversion
304bf98 Create information_session_agenda.md
f018ed8 Create .gitkeep
682ca09 Initial commit
```

Let's go ahead and fetch all our remotes

In [24]:
```
git fetch --all
```

```
Fetching origin
Fetching upstream
remote: Enumerating objects: 1, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (1/1), 927 bytes | 231.00 KiB/s, done.
From github.com:gt-ospo/summer-internship-program
   dcffbc5..5fedbcb  main       -> upstream/main
```

And we can see that `upstream/main` is up-to-date. But notice that `origin/main` and `main` do not have the merge. The PR didn't affect them at all. How do we fix it?

In [26]:
```
git log --remotes --oneline
```

```
5fedbcb (upstream/main) Merge pull request #2 from RonRahaman/module02-train
ing
a2d0a7b (HEAD -> module02-training, origin/module02-training) RR: Correct hy
phen in 'open-source'
dcffbc5 (origin/main, origin/HEAD, main) Update information_session_agenda.m
d
841cca0 Update information_session_agenda.md
c30f9c3 Added VSIP introduction slides
cbf5fcc Delete 2024/project_slides/.gitkeep
aad7b66 Merge pull request #1 from sbryngelson/main
1b64ea5 Update information_session_agenda.md
adb2b7f Update information_session_agenda.md
b0f9827 FL: Updated three slide links
0e769d2 FL: added two more presentation
8a7ddf8 FL: Updte trestle
992f802 Update information_session_agenda.md
b9c8deb Update README.md
a9ba7b5 Update information_session_agenda.md
a2c9156 Update agenda for Summer 2024 session
a7b1b45 Update information_session_agenda.md
7fe5e27 FL: Upload 12 IBM projects with naming conversion
304bf98 Create information_session_agenda.md
f018ed8 Create .gitkeep
682ca09 Initial commit
```

This is one process that just relies on Git (not any GitHub-specific features). As such, it will work on GitHub, Bitbucket, etc. It is described at https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/working-with-forks/syncing-a-fork#syncing-a-fork-branch-from-the-command-line

1. Pull "main" from upstream

In [27]:
```
git checkout main
```

```
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

In [28]:
```
git pull upstream main
```

```
hint: Pulling without specifying how to reconcile divergent branches is
hint: discouraged. You can squelch this message by running one of the follow
ing
hint: commands sometime before your next pull:
hint:
hint:   git config pull.rebase false  # merge (the default strategy)
hint:   git config pull.rebase true   # rebase
hint:   git config pull.ff only       # fast-forward only
hint:
hint: You can replace "git config" with "git config --global" to set a defau
lt
hint: preference for all repositories. You can also pass --rebase, --no-reba
se,
hint: or --ff-only on the command line to override the configured default pe
r
hint: invocation.
From github.com:gt-ospo/summer-internship-program
 * branch            main       -> FETCH_HEAD
Updating dcffbc5..5fedbcb
Fast-forward
 README.md | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

In [29]:
```
git log --oneline --remotes --branches
```

```
5fedbcb (HEAD -> main, upstream/main) Merge pull request #2 from RonRahaman/
module02-training
a2d0a7b (origin/module02-training, module02-training) RR: Correct hyphen in
'open-source'
dcffbc5 (origin/main, origin/HEAD) Update information_session_agenda.md
841cca0 Update information_session_agenda.md
c30f9c3 Added VSIP introduction slides
cbf5fcc Delete 2024/project_slides/.gitkeep
aad7b66 Merge pull request #1 from sbryngelson/main
1b64ea5 Update information_session_agenda.md
adb2b7f Update information_session_agenda.md
b0f9827 FL: Updated three slide links
0e769d2 FL: added two more presentation
8a7ddf8 FL: Updte trestle
992f802 Update information_session_agenda.md
b9c8deb Update README.md
a9ba7b5 Update information_session_agenda.md
a2c9156 Update agenda for Summer 2024 session
a7b1b45 Update information_session_agenda.md
7fe5e27 FL: Upload 12 IBM projects with naming conversion
304bf98 Create information_session_agenda.md
f018ed8 Create .gitkeep
682ca09 Initial commit
```

2. Then push main to origin

In [30]:
```
git push origin main
```

```
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:RonRahaman/summer-internship-program.git
   dcffbc5..5fedbcb  main -> main
```

In [31]:
```
git log --remotes --oneline --branches
```

```
5fedbcb (HEAD -> main, upstream/main, origin/main, origin/HEAD) Merge pull r
equest #2 from RonRahaman/module02-training
a2d0a7b (origin/module02-training, module02-training) RR: Correct hyphen in
'open-source'
dcffbc5 Update information_session_agenda.md
841cca0 Update information_session_agenda.md
c30f9c3 Added VSIP introduction slides
cbf5fcc Delete 2024/project_slides/.gitkeep
aad7b66 Merge pull request #1 from sbryngelson/main
1b64ea5 Update information_session_agenda.md
adb2b7f Update information_session_agenda.md
b0f9827 FL: Updated three slide links
0e769d2 FL: added two more presentation
8a7ddf8 FL: Updte trestle
992f802 Update information_session_agenda.md
b9c8deb Update README.md
a9ba7b5 Update information_session_agenda.md
a2c9156 Update agenda for Summer 2024 session
a7b1b45 Update information_session_agenda.md
7fe5e27 FL: Upload 12 IBM projects with naming conversion
304bf98 Create information_session_agenda.md
f018ed8 Create .gitkeep
682ca09 Initial commit
```

GitHub additionally has a way to sync forks from the web UI:
https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/working-with-forks/syncing-a-fork#syncing-a-fork-branch-from-the-web-ui

In [ ]: