

EPIC GitHub Tutorial (Part 2): Contributing to UFS/EPIC Repositories



Natalie Perlin, NOAA/EPIC, natalie.perlin@noaa.gov

<https://github.com/NOAA-EPIC/training-github>

EARTH PREDICTION INNOVATION CENTER (EPIC)



Welcome!

- This is Part 2 of the EPIC GitHub Tutorial on Contributing to UFS/EPIC Repositories
- View Part 1: <https://epic.noaa.gov/tutorials/?playlist=afde205&video=9e65c0d>
- All the materials and tutorial presentations (Part 1 and Part 2) could be found at
<https://github.com/NOAA-EPIC/training-github>
- Get all the materials and tutorial presentations (Part 1, Part 2):

```
git clone https://github.com/NOAA-EPIC/training-github.git
cd ./training-github
```

- The directory ./training-github/ will contain all of the materials for the tutorial.



Recap of the Part 1: Git and GitHub Basics

- Basic Git and GitHub terms and concepts introduced
- Basic Git configurations and commands discussed
- Example given on setting up a SSH key pair for GitHub authentication
- A new local repository was created, initialized, pushed to GitHub
- Modifications made, committed, local changes tracked, logs viewed
- A new branch made, different branches compared
- Started topic: public GitHub repositories, forks and clones



EARTH PREDICTION INNOVATION CENTER (EPIC)



GitHub Tutorial Part 2: Working with Remote Repositories

- UFS/EPIC Public Repositories, Forks and Clones
- Branches and Tags
- Git Workflow: from Local Spaces to Remote Repos
- Fetching and Merging Remote Branches
- Resolve Merge conflicts
- Making Pull Requests
- GitHub Issues and Discussions



EARTH PREDICTION INNOVATION CENTER (EPIC)



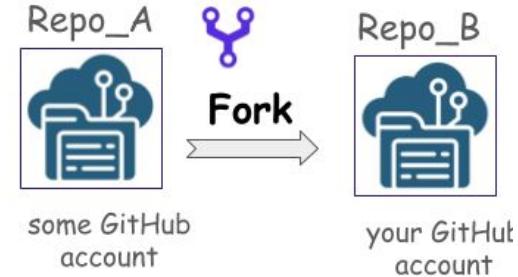
Common GitHub repositories used by UFS community

- <https://github.com/ufs-community/ufs-weather-model>
- <https://github.com/ufs-community/ufs-srweather-app>
- https://github.com/ufs-community/land-DA_workflow
- <https://github.com/ufs-community/uwtools>
- https://github.com/ufs-community/UFS_UTILS
- <https://github.com/ufs-community/ccpp-physics> (forked from [NCAR/ccpp-physics](https://github.com/NCAR/ccpp-physics))
- <https://github.com/hafs-community/HAFS> (forked from [hafs-communityHAFS](https://github.com/hafs-communityHAFS))
- <https://github.com/ufs-community/global-workflow-AR> ([NOAA-EMC/global-workflow](https://github.com/NOAA-EMC/global-workflow))
- <https://github.com/NOAA-EMC/UPP>
- <https://github.com/noaa-oar-arl/NEXUS>
- <https://github.com/NOAA-EMC/AQM-utils>
- <https://github.com/JCSDA/spack-stack>



GitHub Repositories: Forks and Clones

- **fork:** a separate copy of another existing repository
 - created in a remote location, e.g., GitHub account
 - could be synced with the primary repository
 - contribute to your fork not affecting the original repo
- **clone:** a linked local copy of an existing repository
 - a repository code is downloaded to your local machine
 - done through the command ‘git clone’
 - references to an original target/remote repository



EARTH PREDICTION INNOVATION CENTER (EPIC)

GitHub Repositories: Forks and Clones

A screenshot of a web browser showing the GitHub fork interface for the `NOAA-EPIC/training-github` repository. The URL in the address bar is `github.com/NOAA-EPIC/training-github`. The main repository page for `training-github` is visible, showing a commit from `natalie-perlin` updating `README.md` 9 hours ago. On the right side, there is a modal window titled "Existing forks" which displays a message: "You don't have any forks of this repository." Below this message is a button labeled "+ Create a new fork". A red circle highlights the "Fork" button at the top of the modal, and a dashed red circle highlights the "+ Create a new fork" button.

A screenshot of a web browser showing the GitHub repository for `natalie-perlin/training-github`. The URL in the address bar is `github.com/natalie-perlin/training-github`. The repository page shows a commit from `natalie-perlin` updating `README.md` 9 hours ago. At the top of the page, the repository name `training-github` is followed by the text "Public" and "forked from `NOAA-EPIC/training-github`". A large blue arrow points from the left side of the image towards this text. The repository page also includes sections for "About", "Contributors", and "Releases".

EARTH PREDICTION INNOVATION CENTER (EPIC)



GitHub Repositories: Forks and Clones

- **fork:** a separate copy of another existing repository
 - created in a remote location, e.g., GitHub account
 - could be synced with the primary repository
 - contribute to your fork not affecting the original repo
- **clone:** a linked local copy of an existing repository
 - a repository code is downloaded to your local machine
 - done through the command 'git clone'
 - references to an original target/remote repository



GitHub Repositories: Forks and Clones

- **fork:** a separate copy of another existing repository
 - **create** [Natalie@MacbookPro:~]\$ git clone https://github.com/NOAA-EPIC/training-github.git
Cloning into 'training-github'...
remote: Enumerating objects: 16, done.
remote: Counting objects: 100% (16/16), done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 16 (delta 3), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (16/16), 9.95 MiB | 1.52 MiB/s, done.
Resolving deltas: 100% (3/3), done.
 - **cou** [Natalie@MacbookPro:~]\$ cd training-github
 - **con** [Natalie@MacbookPro:~/training-github]\$ git remote -v
origin https://github.com/NOAA-EPIC/training-github.git (fetch)
origin https://github.com/NOAA-EPIC/training-github.git (push)
 - **don** [Natalie@MacbookPro:~/training-github]\$ git branch
* main
 - **refe** [Natalie@MacbookPro:~/training-github]\$
- **clone:** a copy of another existing repository
 - **a re** [Natalie@MacbookPro:~]\$ cd training-github
 - **don** [Natalie@MacbookPro:~/training-github]\$ git branch
* main
 - **refe** [Natalie@MacbookPro:~/training-github]\$



Contributing to an open-source GitHub repository

1. **FORK** the original repository to your GitHub account
2. **CLONE** your repository to a local machine
3. Make changes to your cloned repository
4. **PUSH** them to your GitHub repository: it then will be synced to your changes
5. Create a **PULL REQUEST** (PR) to merge your changes with the original repository: it means making a request to the repository owner and telling them “Please check my changes and merge them if you like it”



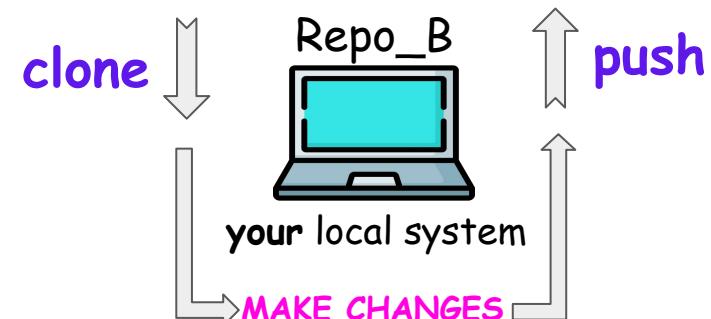
EARTH PREDICTION INNOVATION CENTER (EPIC)



Contributing to an open-source GitHub repository



pull request
←————→
fork



EARTH PREDICTION INNOVATION CENTER (EPIC)

GitHub Repositories: Branches and Tags

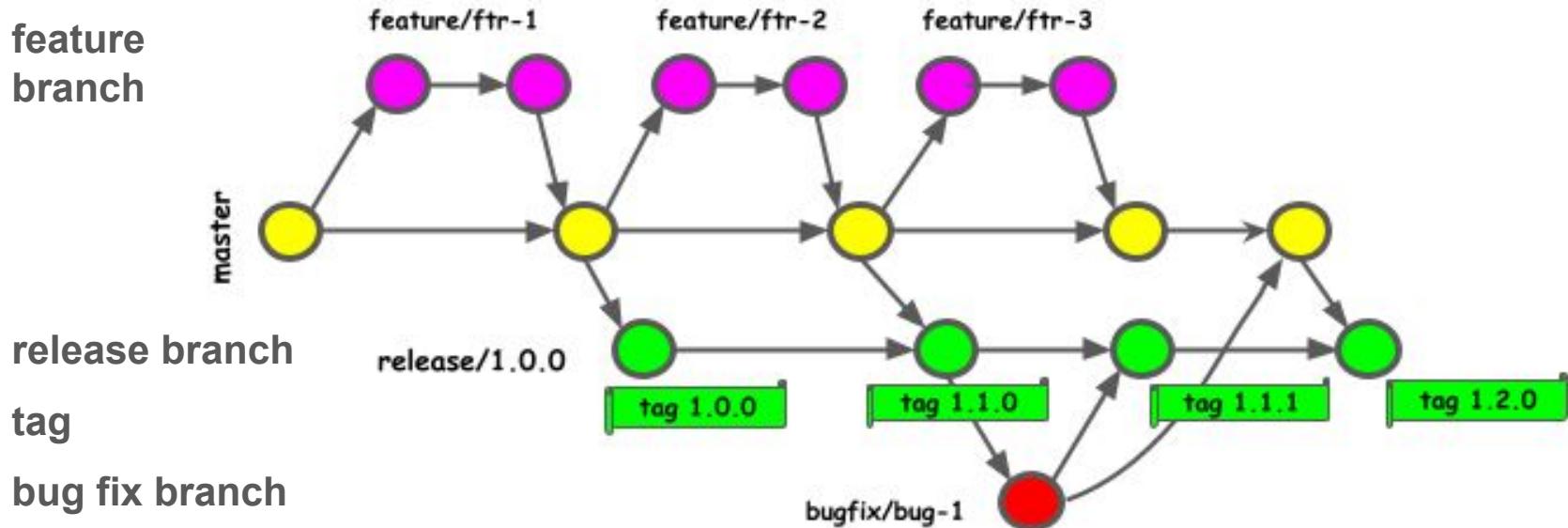
Branches:

- used for concurrent development, bug fixes, implementation *features*
- can be moved or deleted
- often expected to be merged back into original branch
- *feature branches* could have a long span

Tags:

- mark specific points in history to capture software versions
- typically used to mark stable points that correspond to specific releases
- *permanent* and don't change after they are created
- used to label a specific *commit*

GitHub Repositories: Branches and Tags

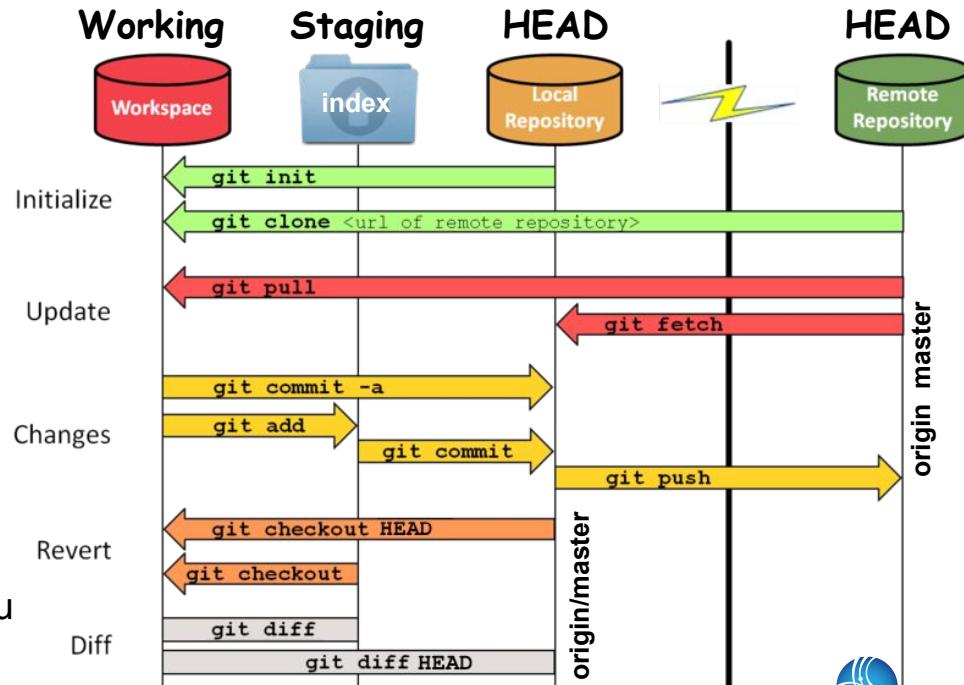


Git workflow: from local spaces to remote repos

Local Git has **three** stages:

- **Working area** -
your local directory,
all untracked, unstaged,
locally deleted files
- **Staging area:**
what needs to be committed,
does not need to be committed,
any new files to be added/removed
- **Local repository (HEAD*):**
committed changes, ,
latest checked out version of the
remote repository

(*) HEAD - symbolic reference to the branch you
are working on, or rather, most recent commit



Git workflow: from local spaces to remote repos

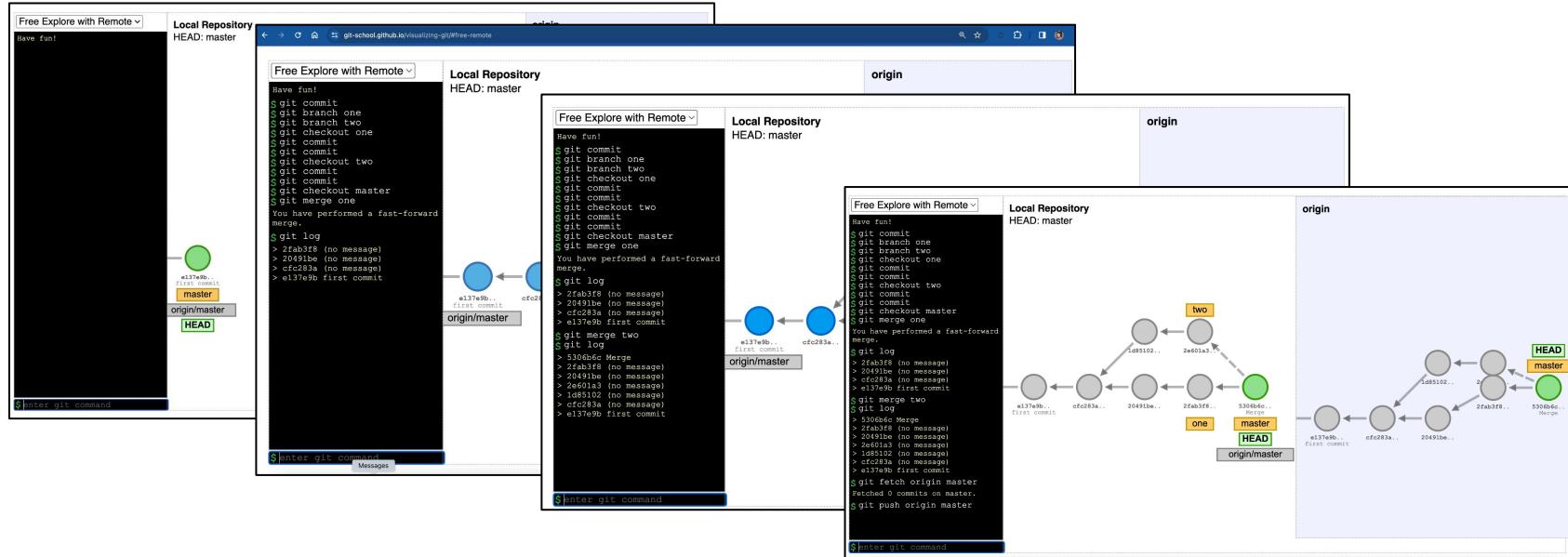
- Use an online workflow visualizer tool: <https://git-school.github.io/visualizing-git>:
 - **case 1:** divergent local branches, merge, update the remote repo
 - **case 2:** divergent local branches, detached HEAD, rebase, merge, bring upstream branch changes, update the remote repo
- Use an existing authoritative repository to test real-case scenarios
 - fetch and merge remote branches, update a fork repository
 - introduce your changes, merge, resolve merge conflicts
 - make a pull request (PR) into the authoritative repository
- Test out a PR, give feedback, make a PR into another developer's code



EARTH PREDICTION INNOVATION CENTER (EPIC)

Visualization cases: case 1 (<https://git-school.github.io/visualizing-git>)

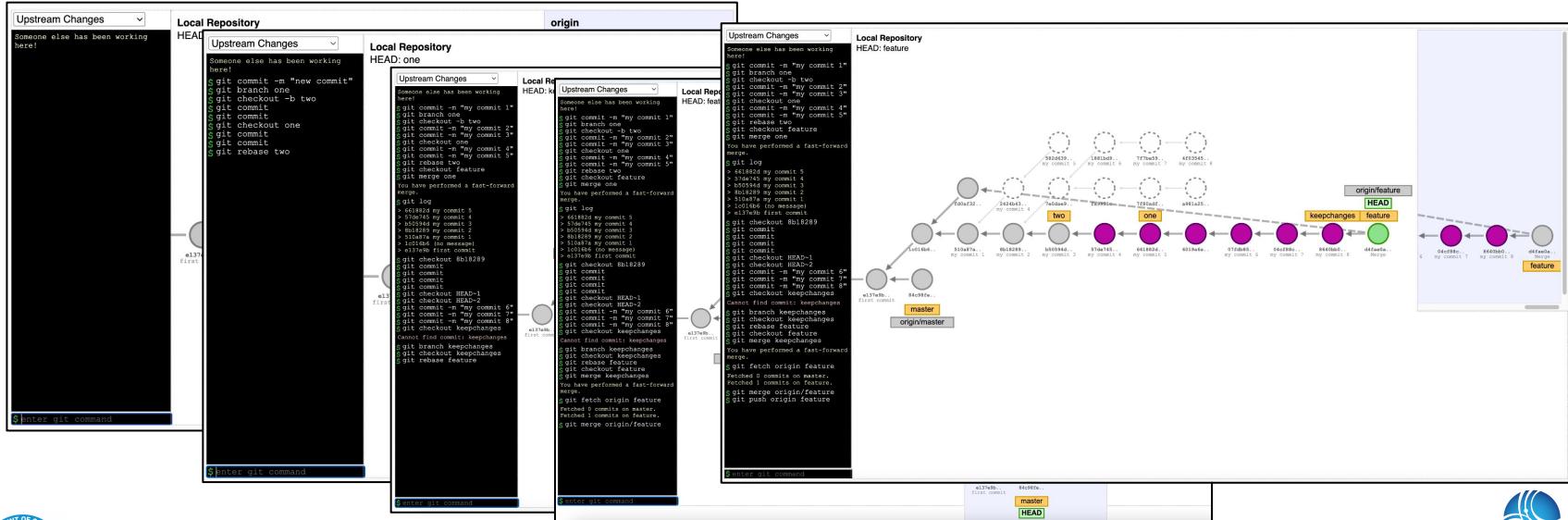
case 1: divergent local branches, merge, update the remote repo



EARTH PREDICTION INNOVATION CENTER (EPIC)

Visualization cases: case 2 (<https://git-school.github.io/visualizing-git>)

case 2: divergent local branches, rebase, detached HEAD, merge, bring upstream branch changes, update the remote repo



General steps to contribute to remote repos

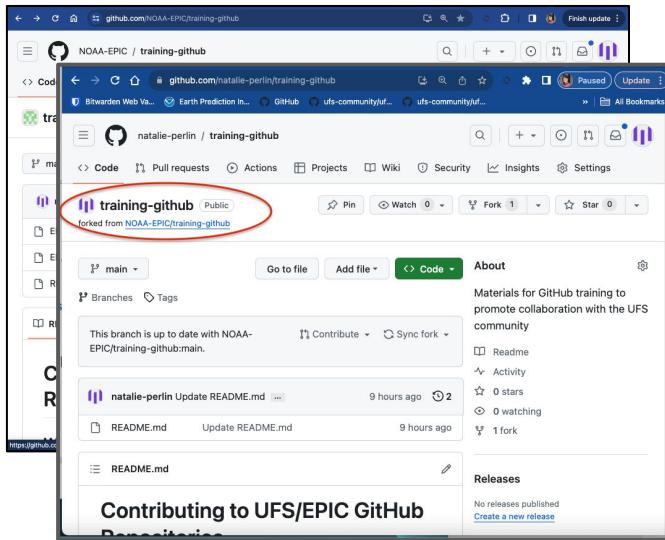
1. Create a fork of the authoritative repo in your GitHub account (dashboard)
2. Clone the repository into your local machine
`git clone <url-of-the-repo-fork>`
3. Add an upstream authoritative repository for the reference and updates
`git remote add upstream <url-of-the-authoritative-repo>`
4. Checkout your repository's main branch: `git checkout master`
5. Fetch any changes from the authoritative repository:
`git fetch upstream`
6. Merge in the changes to the master branch into your working reference
`git merge upstream/master`
7. Push your changes back to GitHub: `git push origin master`
8. Create a pull request (PR) into the authoritative repository (dashboard)



Git workflow example with an existing authoritative repo

An existing repository: <https://github.com/NOAA-EPIC/training-github>

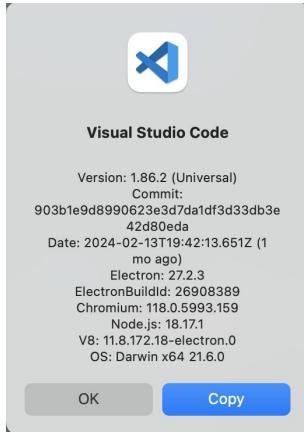
- o fetch and merge remote branches, update a fork repository



```
git clone git@github.com:<repo-fork>
cd <training-github>
git remote add upstream <authoritative-repo>
git remote -v
git checkout master
# make changes, commit them to a master branch
# fetch any updates from the authoritative repository:
git fetch upstream
git merge upstream/master
git log --oneline
git push origin master
```

Git workflow example with an existing authoritative repo

We could use a GUI code editor,
such as Visual Studio Code



The screenshot shows a Mac OS X desktop environment. In the foreground, a terminal window is open with the command 'git log' or a similar command, displaying a list of commits from a GitHub repository named 'training-github'. The commits are listed with their commit hash, author, date, and message. In the background, a code editor window for 'README2' is visible, showing the same five commits as the terminal. The code editor has tabs for 'PROBLEMS', 'OUTPUT', 'TERMINAL', and '...'. The status bar at the bottom of the screen shows the current working directory as '[Natalie@MacbookPro:~/GIT/training-github]\$'.

```
91f65b9 (HEAD -> feature, origin/feature) merged with upstream/master
762ef34 (upstream/master) Create README2
fb61b36 (one) Merge branch 'two' into one
42b2f4d fifth commit
eefd59d fourth commit
d5735b5 (two) third commit
d3c850d second commit
175c441 first commit
4cba347 (origin/master, origin/HEAD, master) Update README.md
daaa25d Added GitHub Tutorial Part 1 files, pdf and pptx
27d98dd Update README.md
68512ca Update README.md
dc91e01 Initial commit
```

Git workflow example 2 with an existing authoritative repo

- An existing authoritative repository: <https://github.com/NOAA-EPIC/training-github>
 - introduce your changes, merge, resolve merge conflicts

```
git clone git@github.com:<repo-fork>
git remote add upstream <authoritative-repo>
git remote -v
git checkout master
git checkout -b feature
# create a file README2, line: # first commit
git add README2
git commit -m "first commit"
git branch one
git checkout -b two
```



Git workflow example 2 (continued)

- An existing authoritative repository: <https://github.com/NOAA-EPIC/training-github>
 - introduce your changes, merge, resolve merge conflicts

```
git branch  
# Add a line to README2 file: # second commit  
git commit -am "second commit"  
# Add a line to README2 file: # third commit  
git commit -am "third commit"  
git checkout one  
# Add a line to README2 file: # fourth commit  
git commit -am "fourth commit"  
# Add a line to README2 file: # fifth commit  
git commit -am "fifth commit"  
git branch  
git merge two
```

```
# solve merge conflicts using a text editor!!  
git commit -am "merge with two"  
git branch  
git checkout feature  
git merge one  
git log --oneline  
git fetch upstream master  
git merge upstream/master  
git checkout upstream/master  
git commit -am "merge with upstream"  
git push origin feature
```



Git workflow example: submit a pull request

- An existing authoritative repository: <https://github.com/NOAA-EPIC/training-github>
 - make a pull request (PR) into the authoritative repository (*)

The screenshot shows the GitHub interface for the repository 'nathalie-perlin / training-github'. The repository is a fork of 'NOAA-EPIC/training-github'. The main page displays recent activity, including a push from 'nathalie-perlin' and several commits from 'EPIC_GitHub_Tutorial'. The 'Compare & pull request' button is highlighted with a red circle.

The screenshot shows the 'Open a pull request' dialog box. It allows users to compare changes between two branches. The 'base repository' is set to 'NOAA-EPIC/training-github' and the 'head repository' is set to 'nathalie-perlin/training-github'. The 'compare' dropdown is set to 'feature:feature'. The dialog also includes fields for adding a title, description, reviewers, assignees, labels, and projects.

(*) - you may need to address any comments, resolve conversations, complete requested changes by PR Reviewers!



EARTH PREDICTION INNOVATION CENTER (EPIC)

Local repository: keep it synched with the remote!

- An existing authoritative repository: <https://github.com/NOAA-EPIC/training-github>
- Remember to sync your primary (master) branch with that from the original repo!

The figure consists of three side-by-side screenshots of a GitHub repository interface. All three screenshots show the same repository: `natalie-perlin / training-github`, which is a fork of `NOAA-EPIC/training-github`.

- Screenshot 1:** Shows the master branch status as "This branch is 1 commit behind NOAA-EPIC/training-github:master". A red circle highlights this message.
- Screenshot 2:** Shows the master branch status as "This branch is 1 commit behind NOAA-EPIC/training-github:master". A red circle highlights this message. Below it, a callout box says "This branch is out-of-date. Update branch to keep this branch up-to-date by syncing a commit from the upstream repository." A red circle highlights the "Sync fork" button, and another red circle highlights the "Update branch" button at the bottom of the callout.
- Screenshot 3:** Shows the master branch status as "This branch is up to date with NOAA-EPIC/training-github:master". A red circle highlights this message.



EARTH PREDICTION INNOVATION CENTER (EPIC)

Git workflow example: testing another user's pull request

- Test out a PR, give feedback, make a PR into another developer's code

```
git clone git@github.com:ufs-community/ufs-srweather-app.git
cd ufs-srweather-app
git pull origin pull/<PR#>/head:PRtest
git checkout PRtest
# test the PR, build, run the code, comment on the PR on specific
# issues, files, or lines
# Make your own PR into original PR author's repository:
# get author's repository location and branch from the PR's page, i.e:
# https://github.com/ufs-community/ufs-srweather-app/pull/1005
```

[develop] Streamline SRW App's interface to
MET/METplus #1005

Open

gsketefian wants to merge 105 commits into [ufs-community:develop](#) from [gsketefian:feature/metplus_conf_templates](#) 

Edit

< > Code ▾



Perusing Public Repositories

- Check out a few different repositories with submodules:

```
git clone --recursive <url-repo>
```

```
git clone --recurse-submodules <url-repo>
```

```
git clone <url-repo>
```

```
cd <repo>
```

```
git submodule update -remote
```

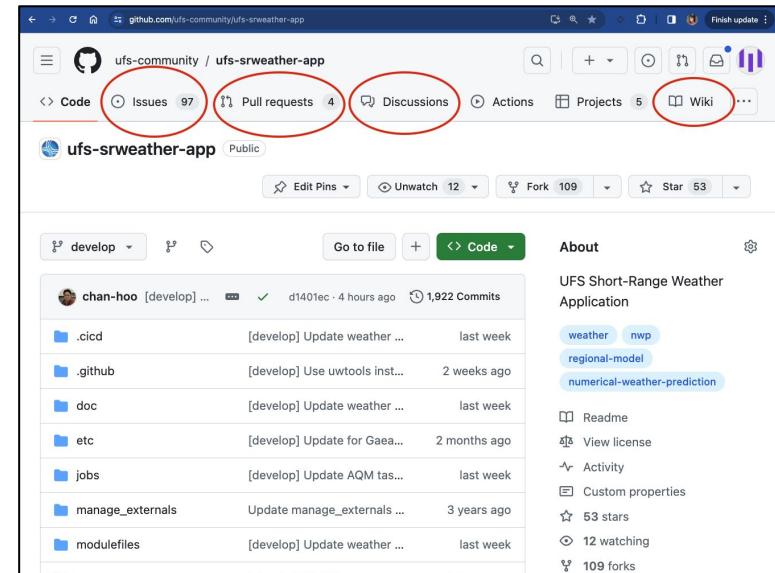
```
git clone --recurse-submodules https://github.com/jcsda/spack-stack.git
```

```
git clone --recursive https://github.com/ufs-community/ufs-weather-model.git
```



Perusing Public Repositories

- Submitting issues - “**Issues**”
- Reaching out for help through discussions - “**Discussions**”
- Checking out and testing existing PRs - “**Pull Requests**”
- Navigating documentation - “**Wiki**”, look for links to **readthedocs.io**, such as
<https://ufs-weather-model.readthedocs.io/>
<https://ufs-srweather-app.readthedocs.io/>



EARTH PREDICTION INNOVATION CENTER (EPIC)



Perusing Public Repositories

The image shows three screenshots of a GitHub repository interface for the `ufs-community / ufs-srweather-app` repository.

- Screenshot 1:** Shows the Issues page with 97 open issues. A search bar at the top has the query `is:issue is:open`. On the left, a sidebar lists several pull requests with green checkmarks, such as `[SRW-AQM] Need to`, `[SRW-AQM] Need to`, and `[develop] Port SRW-AQM to C`.
- Screenshot 2:** Shows the Pull requests page with 4 open pull requests. A search bar at the top has the query `is:pr is:open`. A red circle highlights the "New discussion" button in the top right corner of the pull request list.
- Screenshot 3:** Shows the Home page of the repository. It features a prominent announcement banner: "Announcement: UFS Short-Range Weather Application 10/31/2023". Below the banner, there's a "Welcome to the UFS Short-Range Weather (SRW) Application Wiki!" section. This section includes a paragraph about the SRW App v2.2.0 release, links to the User's Guide and NOAA EPIC website, and a detailed description of the repository's purpose and the forecast model used. To the right, there's a sidebar with navigation links like "Home", "Supported Platforms and Compilers", "Getting Started", "Releases", and "Announcements".

EARTH PREDICTION INNOVATION CENTER (EPIC)

