Questions?

Model Development

- Creating a GitHub Fork
- Working with multiple remotes in a git clone
- Model development – commits and pushes
- Adding a new model (e.g., BLOM, CAM, CLM) to NorESM.

Model Development – Working with the code

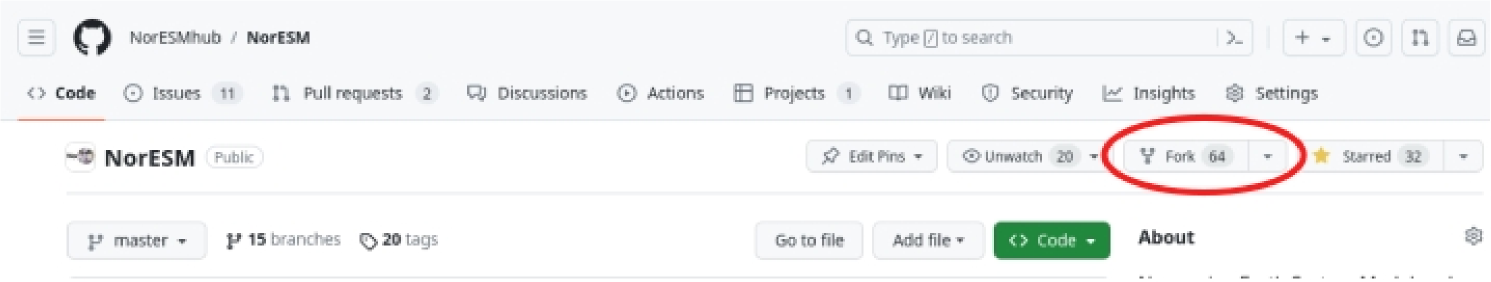
Steps to working with code modifications using git and GitHub

1. Create a personal fork of the repository where you want to make changes*
2. Add that fork as a new remote in your clone*
3. Create and checkout a branch to store your changes
4. Make any changes to your branch and run tests (experiments)
5. Commit early and often to make it easy to track what was changed when
6. Push your branch to your fork for backup and sharing

* One time set-up

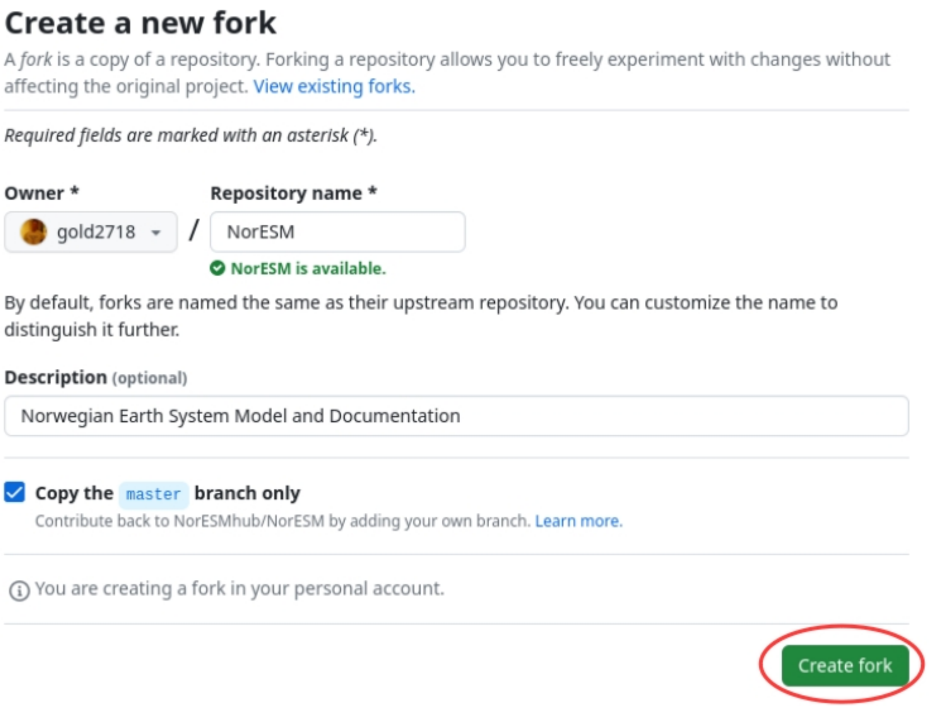
Working with the code – create a personal fork

- A “fork” is a GitHub term for a repository that remembers where it came from.
- A fork is a standard GitHub repository with the added feature that you can see the other forks and open a Pull Request (PR) to one of them.
- From the main page of NorESMhub/NorESM (or NorESMhub/<component>):



Working with the code – create a personal fork

- After you are satisfied with with the settings, press the “Create Fork” button.



Create a new fork

A *fork* is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. View existing forks.

Required fields are marked with an asterisk (*).

Owner * gold2718 / **Repository name *** NorESM

✓ NorESM is available.

By default, forks are named the same as their upstream repository. You can customize the name to distinguish it further.

Description (optional)

Norwegian Earth System Model and Documentation

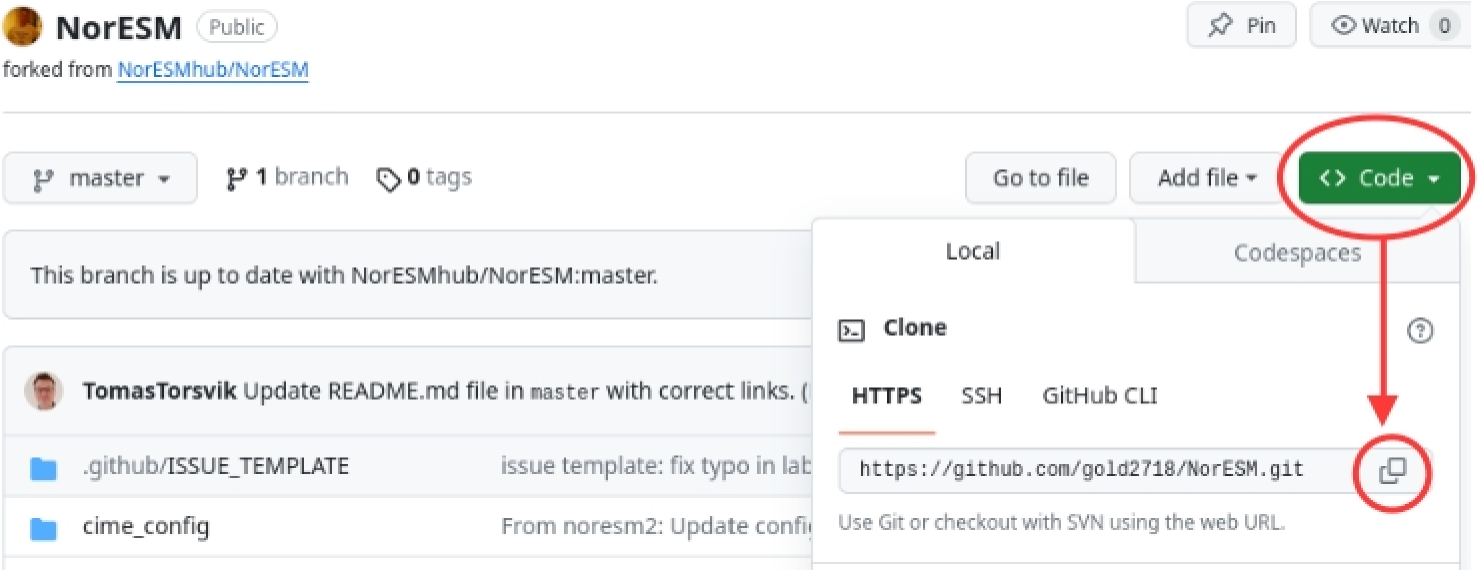
☑ **Copy the `master` branch only**

Contribute back to NorESMhub/NorESM by adding your own branch. Learn more.

ⓘ You are creating a fork in your personal account.

Working with the code – adding a new remote

- Copy the URL for your new fork



NorESM Public

forked from NorESMhub/NorESM

This branch is up to date with NorESMhub/NorESM:master.

TomasTorsvik Update README.md file in `master` with correct links. (

.github/ISSUE_TEMPLATE — issue template: fix typo in lab

cime_config — From noresm2: Update confi

Clone

```
https://github.com/gold2718/NorESM.git
```

Use Git or checkout with SVN using the web URL.

Working with the code – adding a new remote (cont.)

- Copy the URL for your new fork

```
$ git remote add <name> https://github.com/<name>/NorESM.git
```

```
$ git remote -v
```

```
origin https://github.com/NorESMhub/NorESM (fetch)
origin https://github.com/NorESMhub/NorESM (push)
<name> https://github.com/<name>/NorESM.git (fetch)
<name> https://github.com/<name>/NorESM.git (push)
```

Working with the code – creating a new branch

- A branch in git is just a new pointer to the last commit.
- When creating a new branch, always declare where the branch will begin

```
$ git branch my_edit release-noresm2.0.7

$ git checkout my_edit
```

- You can combine the new branch creation and checkout

```
$ git checkout -b my_edit release-noresm2.0.7
```

Working with the code – making code changes

- If you make changes to source code, you need to call **`./case.build`** again
- If you change any Fortran **`use`** statements, you should rebuild the model:

```
./case.build –clean <component>; ./case.build
```

- where **`<component>`** is a component type such as **atm** or **ocn**.
- The fastest way to force a rebuild is: **`rm -r bld`**
- If you change anything in **cime_config**, you should start with a new case

Working with the code – using commits as a development tool

- Make sure your git environment is configured correctly
- For any machine you work on (e.g., Betzy), you need to have these settings:

```
git config --global user.name "Your Name"

git config --global user.email <GitHub email address>
```

- Useful introduction to git from Software Carpentry: https://swcarpentry.github.io/git-novice/
- Also, see the “contribute” section in the NorESM documentation
- If you like looking at code differences graphically, you can try git’s difftool:

```
git config --global diff.tool=meld

git config --global difftool.prompt=false
```

Working with the code – using commits as a development tool

- Commit often! Commits help you track differences and find problems.
- Always enter a useful log message, future you will thank you.
- If you create a new source file, be sure to add it before making your commit:

```
$ git add <new_file>
```

- To commit all changes and type in a log message:

```
$ git commit -am ‘Great log message here’
```

- To commit all changes and have an editor window open to enter your log message:

```
$ git commit -a
```

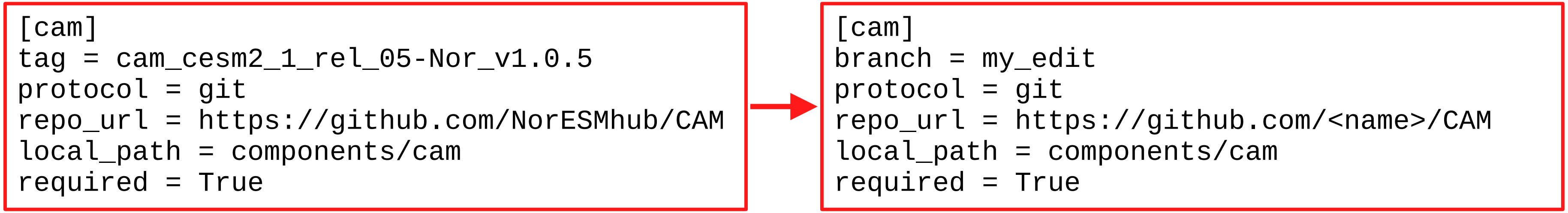
Working with the code – interacting with your fork

- Push your branch to your fork. This serves both as a backup and as a means to share – with others or with yourself on other machines.
- Always specify the destination repository and the branch that you want to push.

```
git push <name> my_edit
```

Working with the code – updating Externals.cfg

- If you are making edits to a branch of a component model and want to perform coupled runs with NorESM, you need to update the Externals.cfg file in NorESM. (Note that for most component models (e.g., CAM, CTSM), you can build and run non-coupled cases with just a clone of that component.)
- Make a branch of your NorESM clone and edit Externals.cfg (example below for CAM):



```
[cam]
tag = cam_cesm2_1_rel_05-Nor_v1.0.5
protocol = git
repo_url = https://github.com/NorESMhub/CAM
local_path = components/cam
required = True
```

```
[cam]
branch = my_edit
protocol = git
repo_url = https://github.com/<name>/CAM
local_path = components/cam
required = True
```

Questions?