Using Git and GitHub with R

Kevin Shook and Rob Chlumsky

June 3, 2020

CSHS SCSH

Canadian Society for Hydrological Sciences

Société Canadienne des Sciences Hydrologiques

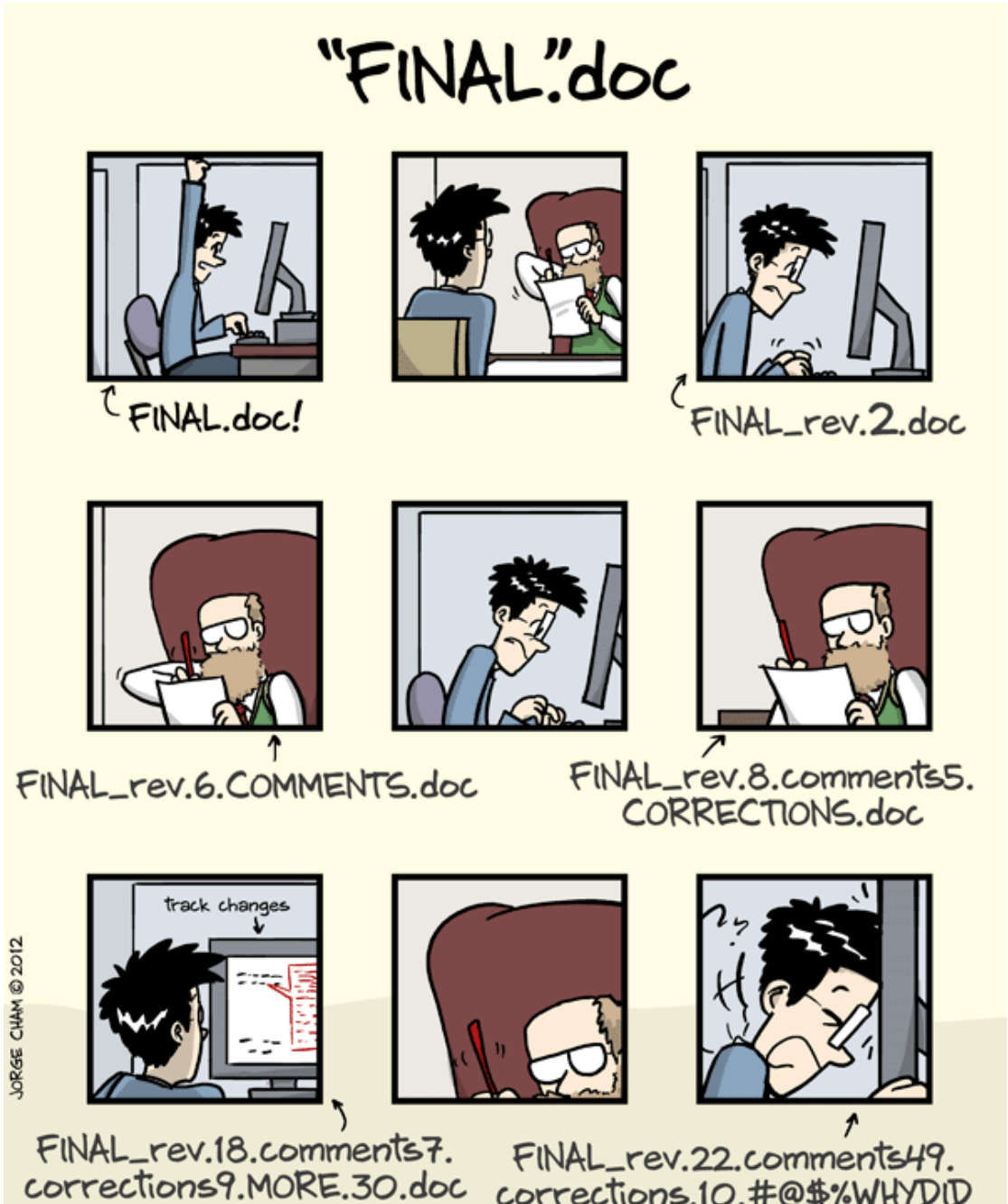
Canadian Water Resources Association

Association Canadienne des Ressources Hydriques

Outline

- Version control
- What are git and GitHub?
- How to set up
- Using git in R
- Working with GitHub

Why version control?



Version control programs

- When you create R files (code, notebooks, documents), there are always changes
- Changes sometimes damage the files
  - need to go back to older versions
- Need to add new features without damaging current version
- Especially true when working with other people
- Version control programs allow you to manage the versions of the files that you create.

git

- Most popular version control program
- Written by Linus Torvalds, creator of Linux
- Free Open Source Software (FOSS)
- *Distributed* version control
  - doesn't require a centralised server like SVN

GitHub

- Website running git
- Allows you to backup your git repository
- Also allows collaboration with others
- There are other similar sites like GitLab: https://about.gitlab.com/

Getting git

- Built into Linux
- For MacOS or Windows, you can download git from https://git-scm.com/

How git works

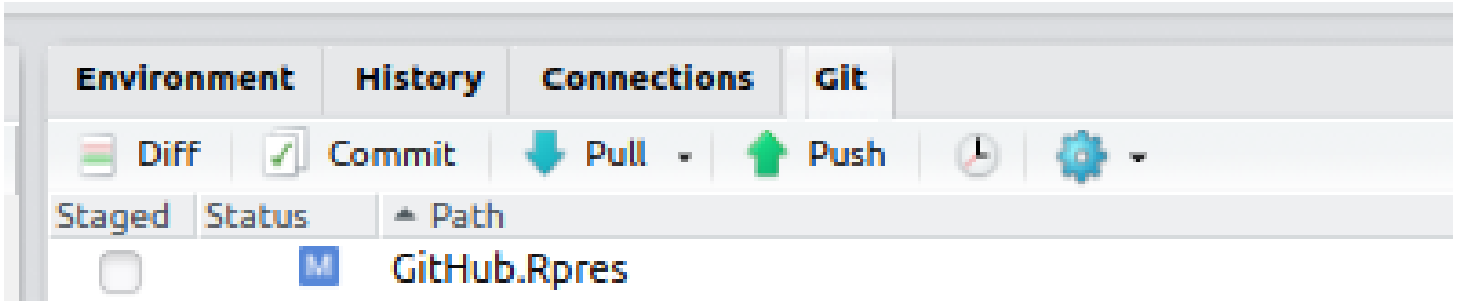
- A folder called **.git** is created in the directory holding your your project, the working directory
- This is the repository
  - It contains all versions, current and old, of your files
- When you make changes to the files, you add them to the repository
- You can retrieve old versions of the files into the working directory

git and Rstudio

- When you clone a repository from GitHub, a local repository is automatically created
- You can also set up git for a local project in Rstudio using the menu
  Tools | Project Options ...

Working with git

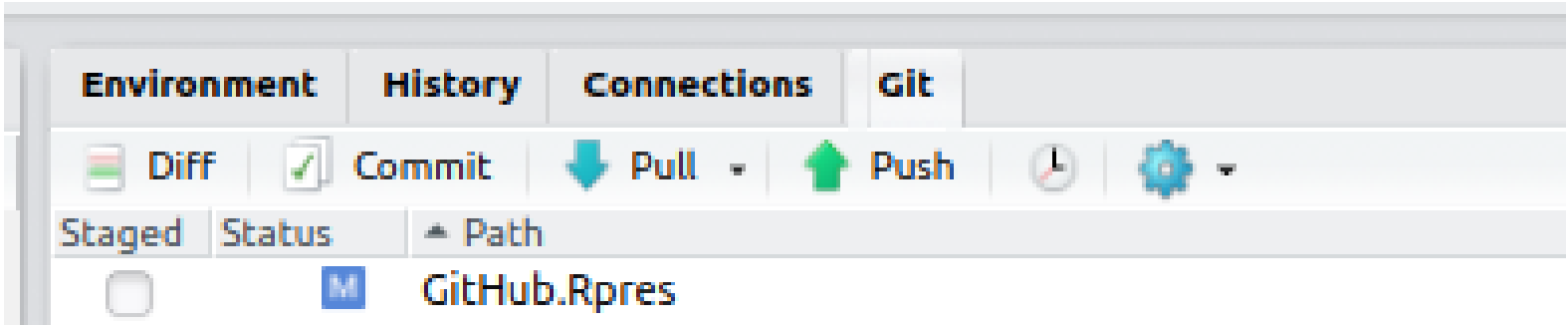
- git is a command-based program
- There are many GUIs for git, including Rstudio
  - makes working with git much easier
  - uses Git tab in top-right



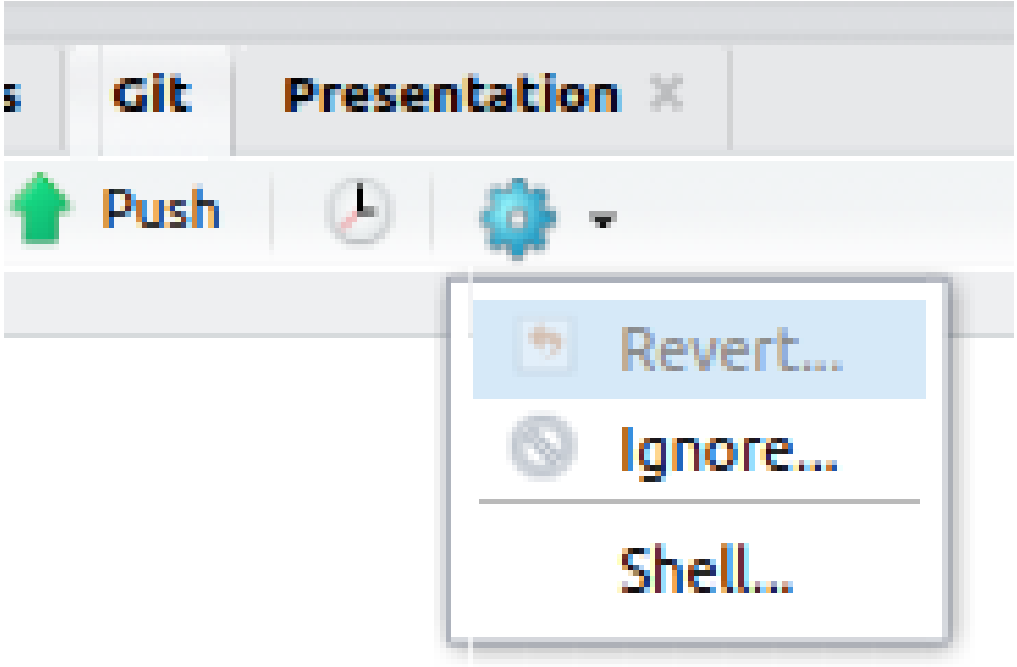
  - you will still have to type commands occasionally

Typing in commands

- In Linux or Mac OS, you can type in git commands in any terminal, including the terminal tab in Rstudio



- In Windows, you have to use the git shell, which is accessed through a drop-down menu



Configuring git

- The first step is to tell git who you are:

```
git config --global user.name "John Doe"
git config --global user.email johndoe@example.com
```

- You can list your current settings with the command

```
git config --list
```

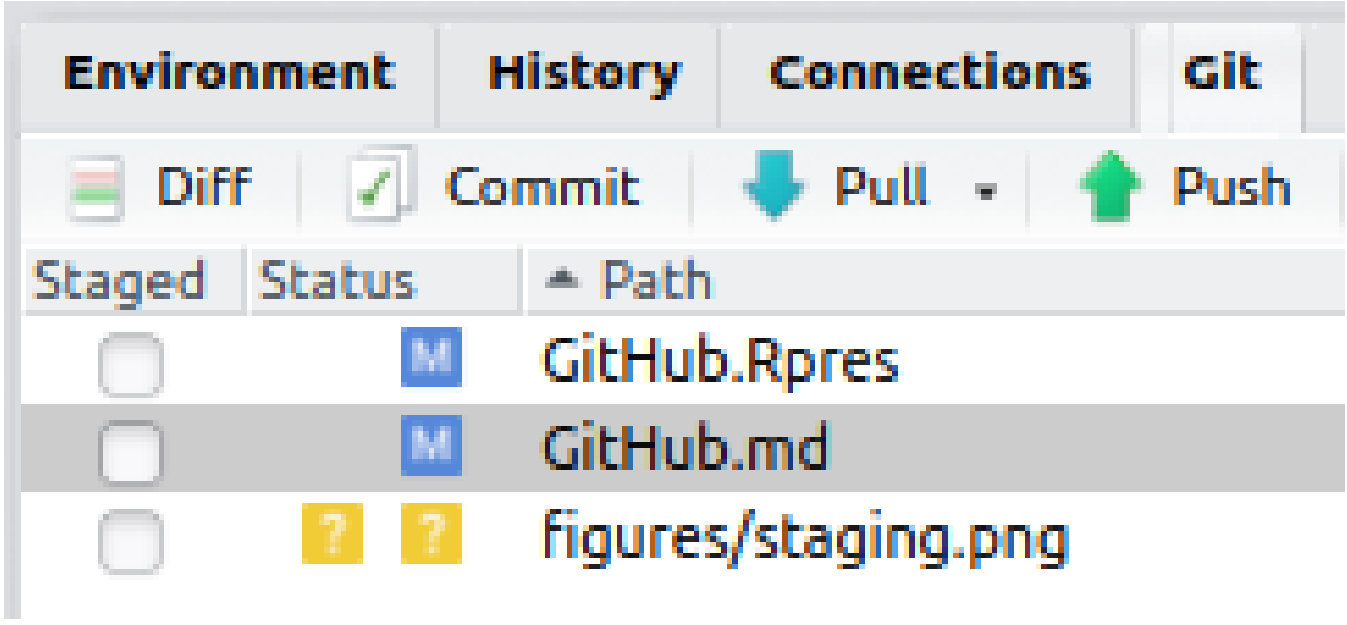
Version control

- As you create code, you will want to add it to the repository
  - generally each time you have made a significant change to any file
- Adding takes 2 steps:

1. Staging (selecting the files to add), and
2. Committing (adding the files to the repository)

Files available for staging

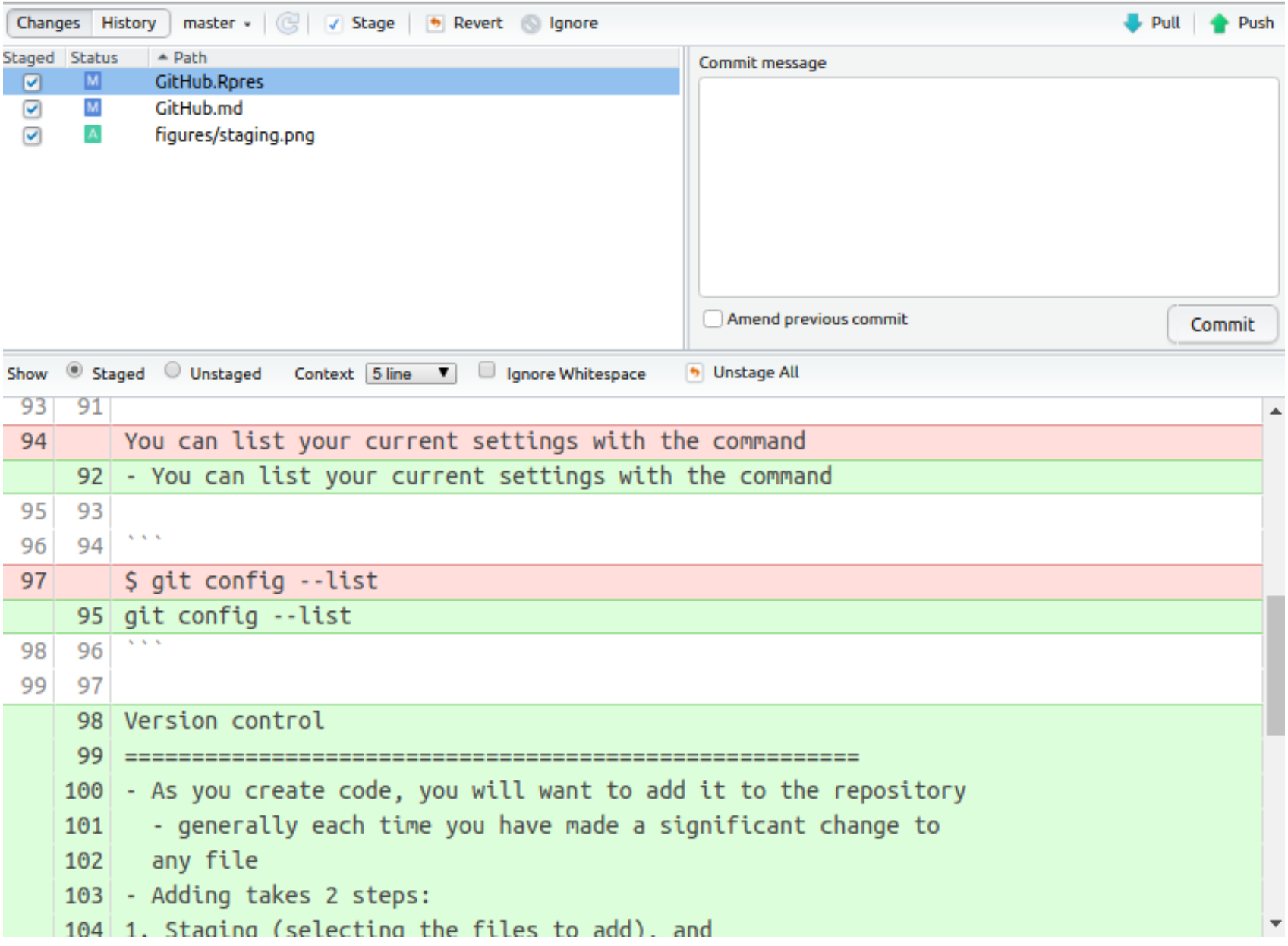
- The Git tab shows all of the files which can be staged
- 2 files have been modified (blue M icon),
- 1 file is new (yellow ? icon)



Committing

- Select the files to be added
  - the icons of the new files will change)
  - and click on the commit icon
- The commit window will pop-up, giving you a chance to review the files before committing

Commit window

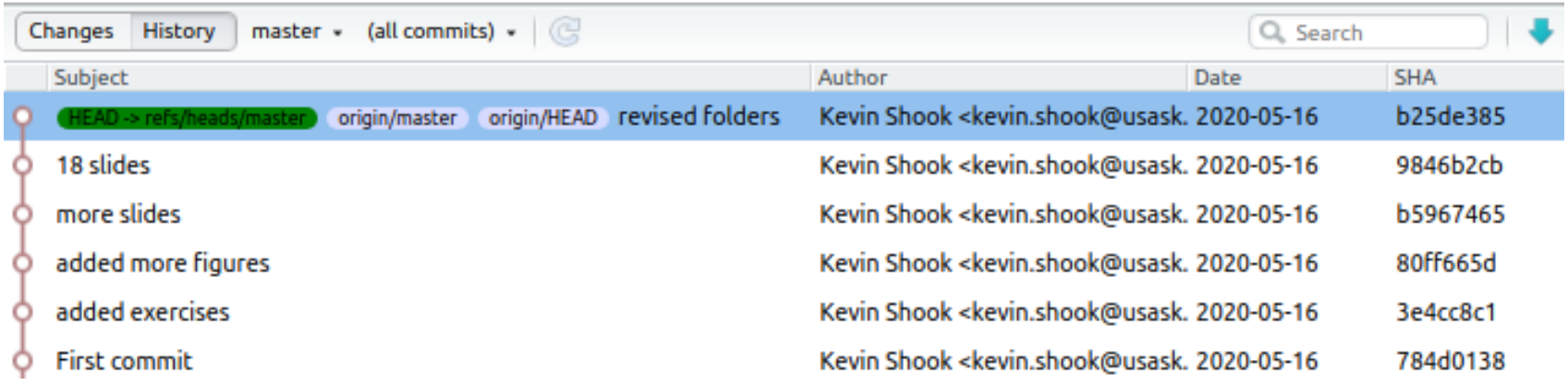


Committing

- The bottom pane (Diff) shows the changes in all of the files
  - you can select or discard changes
- You **must** add a comment in the top-right panel before clicking on Commit

Git history

- In the Commit window, clicking on the History button shows the history of all of your commits to the repository



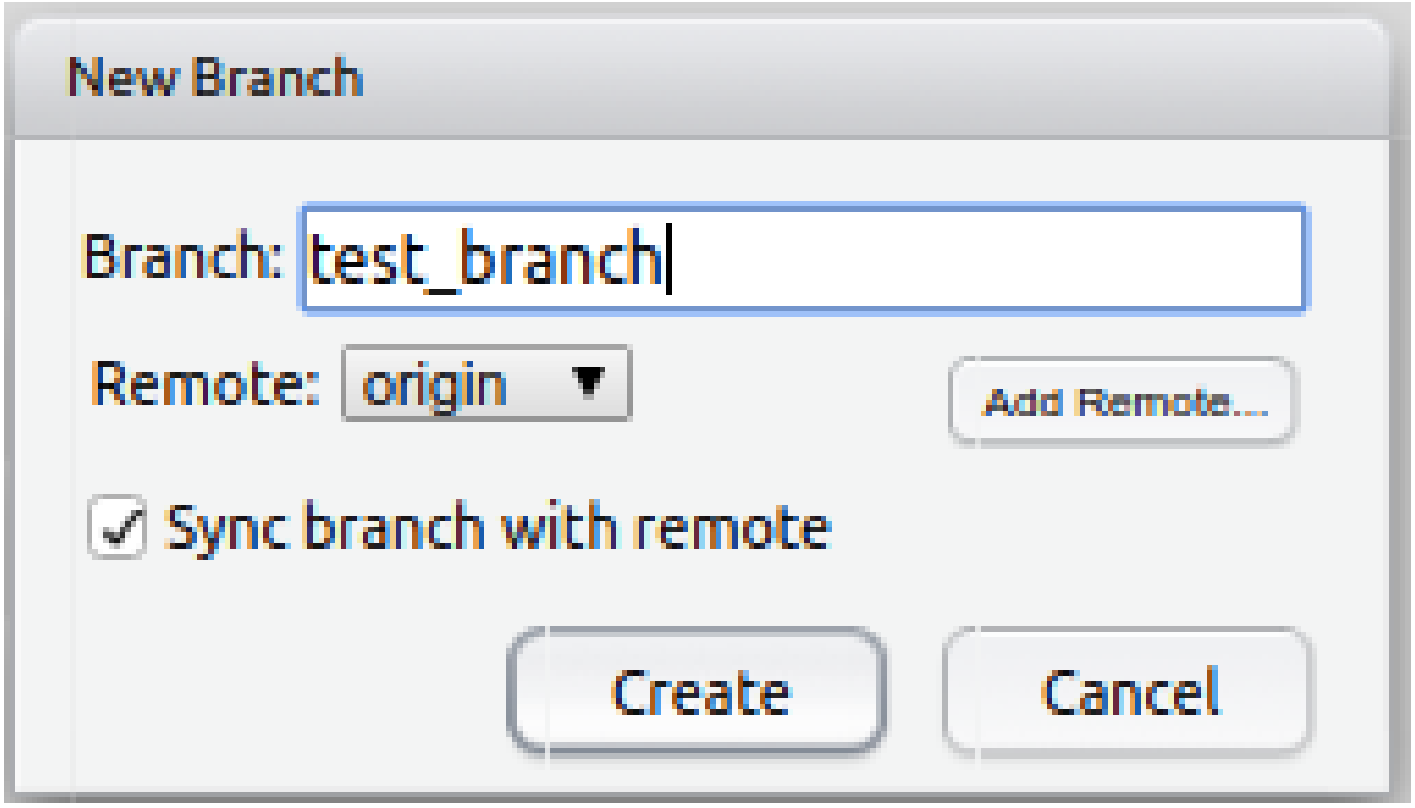
- Each commit is identified by a unique SHA number

Branches

- git uses *branches* to organise your code/documents
- Each repository always has a brance called **master**
  - most up-to-date, best version of the code
- Each branch is separate, and can be changed/deleted
- The current branch is shown in the Git tab
- You can add branches at any time
- When you change the branch, the files in the working directory are updated

Creating branches

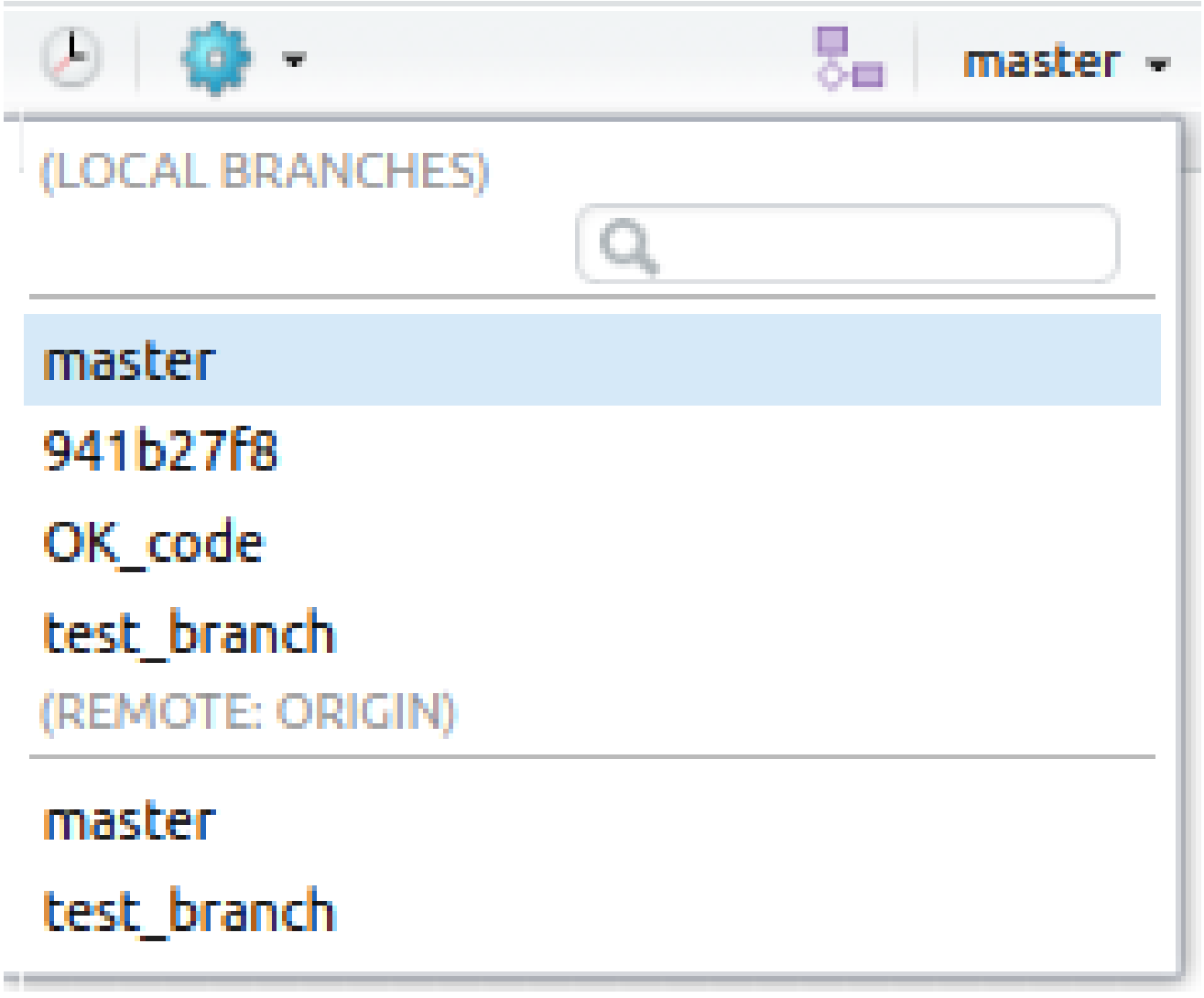
- You can create a new branch at any time
  - Use the branch icon in RStudio:  to display the dialog box



- Current versions of all files are added to the new branch

Changing between branches

- You can switch between branches by selecting the branch name



Recovering from a mistake

- There are lots of ways of screwing up your code
  - you might need to undo a commit
- Fortunately, you can recover old commits

This web site is very useful as it shows many different situations and how to recover from them:

https://sethrobertson.github.io/GitFixUm/fixup.html#committed

Working with GitHub

- The GitHub/GitLab repository linked to your local repot is referred to as the “Remote”

Pulling

- Pulling downloads the GitHub repo to your local repo
- It's a good idea to click on Pull to make sure that the local repo is up to date before doing any new work

Git Pull

```
>>> git pull
Already up to date.
```

Pushing

- Pushing uploads your local repository to GitHub
  - You should only push to your *own* GitHub repository

Git Push

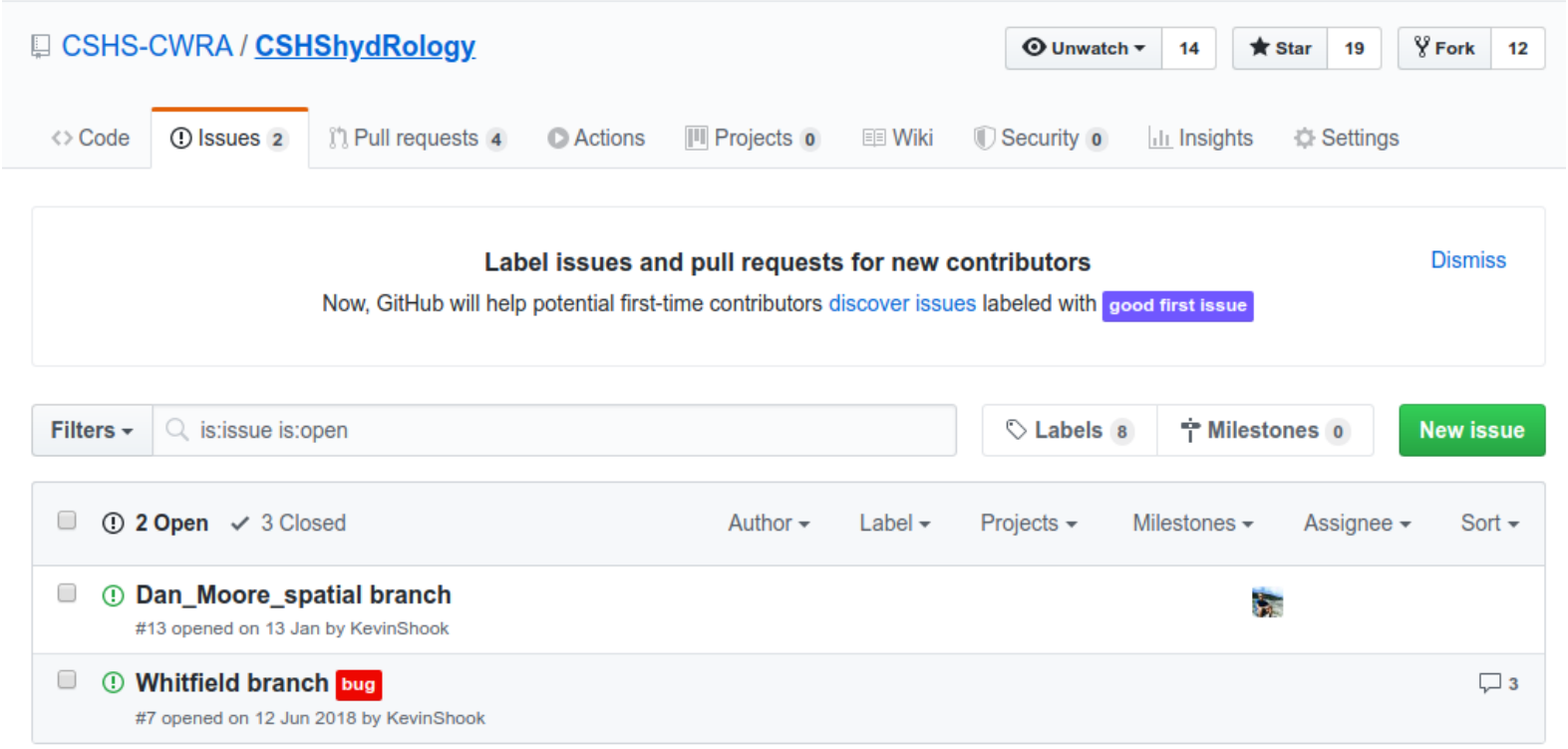
```
>>> git push origin HEAD:refs/heads/master
To github.com:CentreForHydrology/git_for_R.git
   3e4cc8c..b596746  HEAD -> master
```

Working with others

- The most important feature of GitHub is the way it enables people to work together on projects
- Each project will typically have an owner, and one or more people who can approve changes
- If you aren't one of these people (and even if you are!), you shouldn't be pushing changes to the **master** branch directly.

Bug reports

- One of GitHub's most important features.
- Very easy to submit an Issue



- Writing a *good* bug report is an art - see https://github.com/rstudio/rstudio/wiki/Writing-Good-Bug-Reports

Forking

A *fork* is complete copy of a GitHub repo
- It lets you copy other work to use as a basis for your own
- It also lets you make a working copy the repo files, without affecting the original repo - A good way to create new features or fix bugs - When you are finished, you can then submit a Pull Request

Pull requests

ssh

- ssh is short for “secure shell”
- provides secure, encrypted communication between 2 computers
- if you set it up on your computer, you can avoid having to type in your user name and password every time
- part of Linux and Mac OS
- to add to Windows https://jcutrer.com/windows/install-openssh-on-windows10