Git Workshop

Poorani Subramanian, Ph.D.

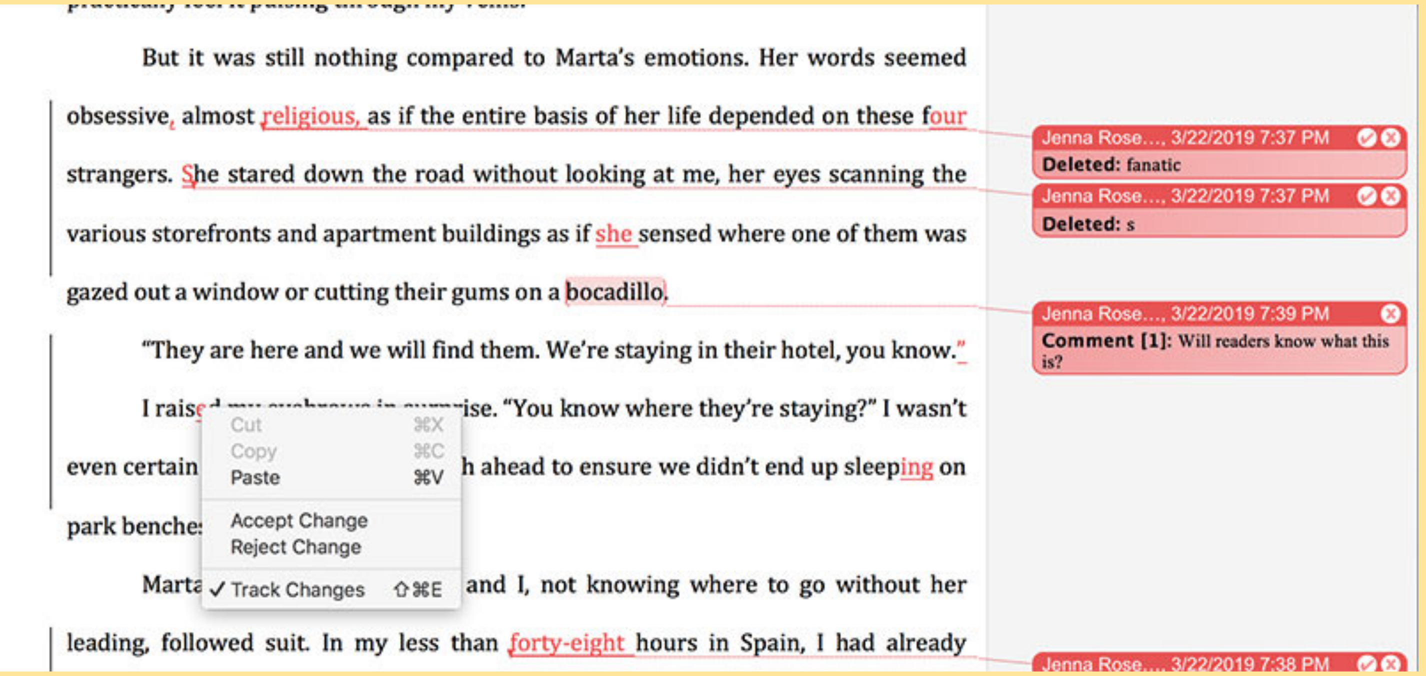
ASBCB ‘Omics Codeathon – 2 October 2022

Key Concepts

1. Version Control
2. Git
3. GitHub

What is Version Control?

- A system of managing a set of files – keeping track of their versions
- VCS you might already be using:
  - Word processor track changes
  - Cloud storage versions (Google Drive, Dropbox, etc)



But it was still nothing compared to Marta's emotions. Her words seemed
obsessive, almost religious, as if the entire basis of her life depended on these four
strangers. She stared down the road without looking at me, her eyes scanning the
various storefronts and apartment buildings as if she sensed where one of them was
gazed out a window or cutting their gums on a bocadillo.

"They are here and we will find them. We're staying in their hotel, you know."

I rais ise. "You know where they're staying?" I wasn't
even certain h ahead to ensure we didn't end up sleeping on
park benche

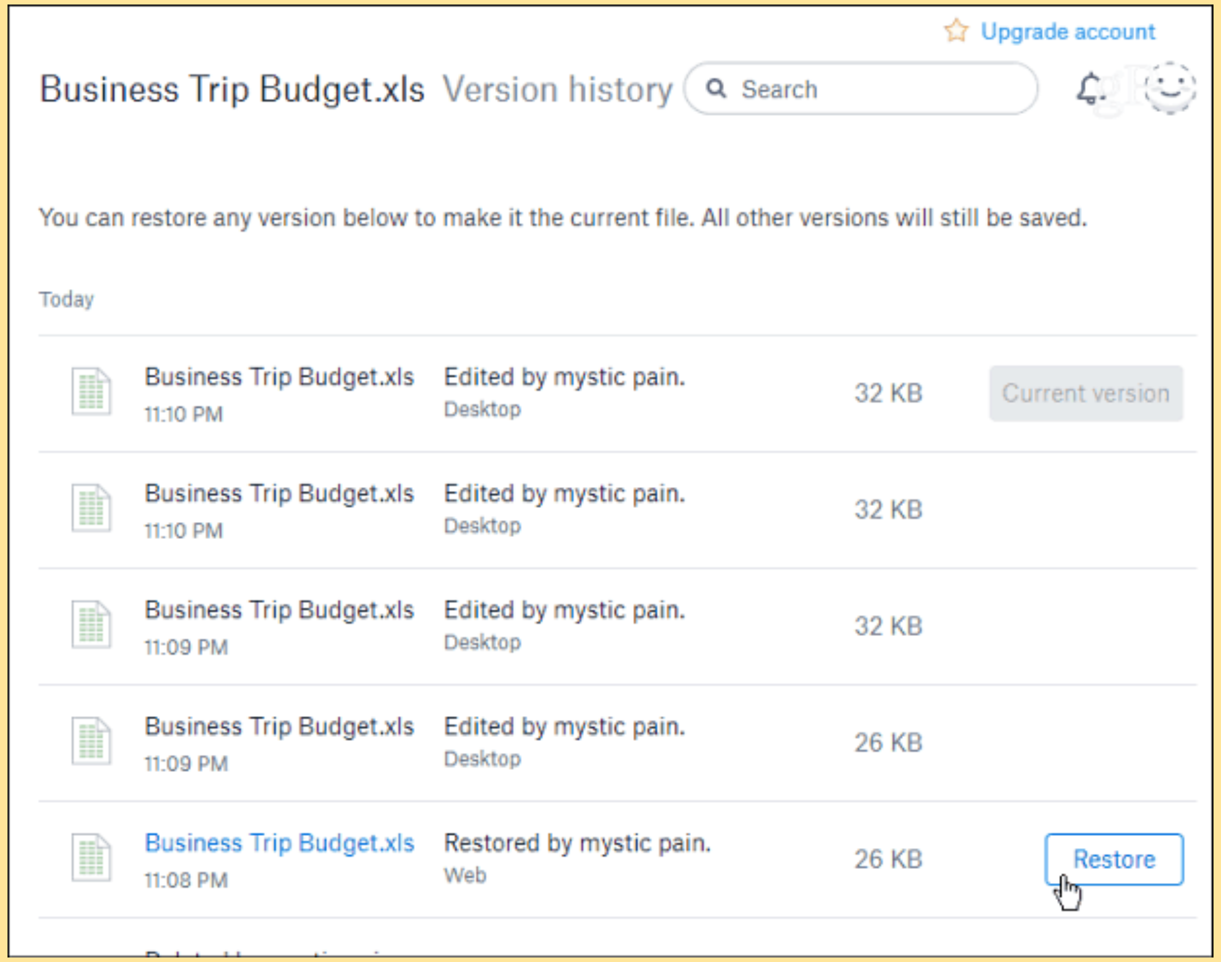
Marta and I, not knowing where to go without her
leading, followed suit. In my less than forty-eight hours in Spain, I had already

Jenna Rose..., 3/22/2019 7:37 PM
Deleted: fanatic

Jenna Rose..., 3/22/2019 7:37 PM
Deleted: s

Jenna Rose..., 3/22/2019 7:39 PM
Comment [1]: Will readers know what this is?

Jenna Rose..., 3/22/2019 7:38 PM



Business Trip Budget.xls Version history

You can restore any version below to make it the current file. All other versions will still be saved.

Today

Business Trip Budget.xls 11:10 PM	Edited by mystic pain. Desktop	32 KB	Current version
Business Trip Budget.xls 11:10 PM	Edited by mystic pain. Desktop	32 KB	
Business Trip Budget.xls 11:09 PM	Edited by mystic pain. Desktop	32 KB	
Business Trip Budget.xls 11:09 PM	Edited by mystic pain. Desktop	26 KB	
Business Trip Budget.xls 11:08 PM	Restored by mystic pain. Web	26 KB	Restore

> **Common features of VCS**
>
> 1. Track changes to a file
> 2. Create checkpoint (version) for a file state
> 3. Revert to an older version of a file (rollback)
> 4. Ignore files if you don’t want to track changes to them

- Advantages of using VCS:

1. Compare the changes made to a file
2. Revert the project to an older state
3. See who made what changes to the files
4. Recover files if you lose them by mistake**
5. Collaborate with others

Popular VCS

Especially for Software development

- Git and subversion (svn) are the most popular
- Svn was very dominant maybe 15-20 years ago
  - R programming language
  - Linux kernel

TOP 5
Version Control
Systems used today
mercurial
SUBVERSION
Bazaar
git
CVS

What is Git?

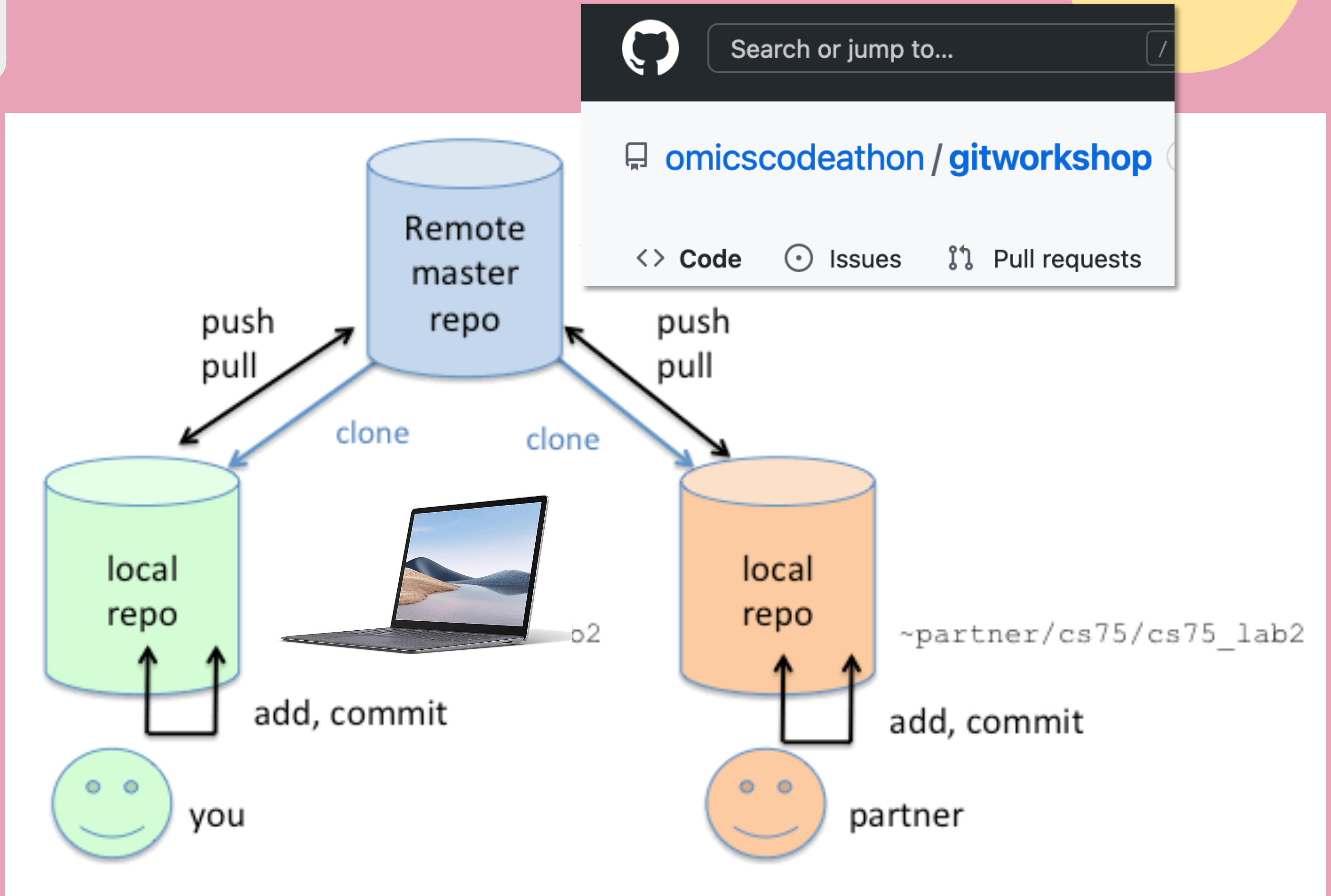
- Git is a particular VCS program
- Mostly used for software development, but could be used to track and manage most files**
- Open source and free program
- Written in C with some shell & Perl – for the command line, and has an API in many languages
- Where do I get git?
- https://git-scm.com/downloads
- Package manager – some OS include it (macOS and some flavors of Linux)

What is GitHub?

- GitHub is a web application that uses git extensively under the hood
- It hosts ***remote repositories***
- There are other web apps:
  - Gitlab.com
  - Bitbucket.com

Simple Workflow

1. *Clone* a *remote* repo
2. *Add* a file
3. *Commit* (save) the changes
4. *Pull* any changes that might have happened while we were working
5. *Push* our changes



How big of a change should I commit?

- The changes in a commit should all relate to each other
  - They should aim to solve a single, small, discrete problem
- If they are unrelated, then rolling back to different versions becomes tricky if just part of the commit is bad or good
- Sometimes useful to break a problem/task into smaller steps and then commit each of those separately
- In general, more small commits are better than fewer, big ones

What kinds of files can I keep in git?

- Git does not track files *per se.* It tracks ***the content of those files***
- It stores the incremental changes that you make
- OR IT TRIES TO
- It is easy to track changes for plain text files – it just has to save the particular lines that have changed – this is more efficient than saving multiple copies of the whole file.
- This is harder to do for binary blobs – png files, etc. It may just keep multiple copies of the file ☹. GitHub and git have improved a little in this regard
- *But you should still not commit binary-type files that will be changing often*, esp if you share the repo with others – as it may make the repo VERY LARGE

What kinds of files should I keep in GitHub?

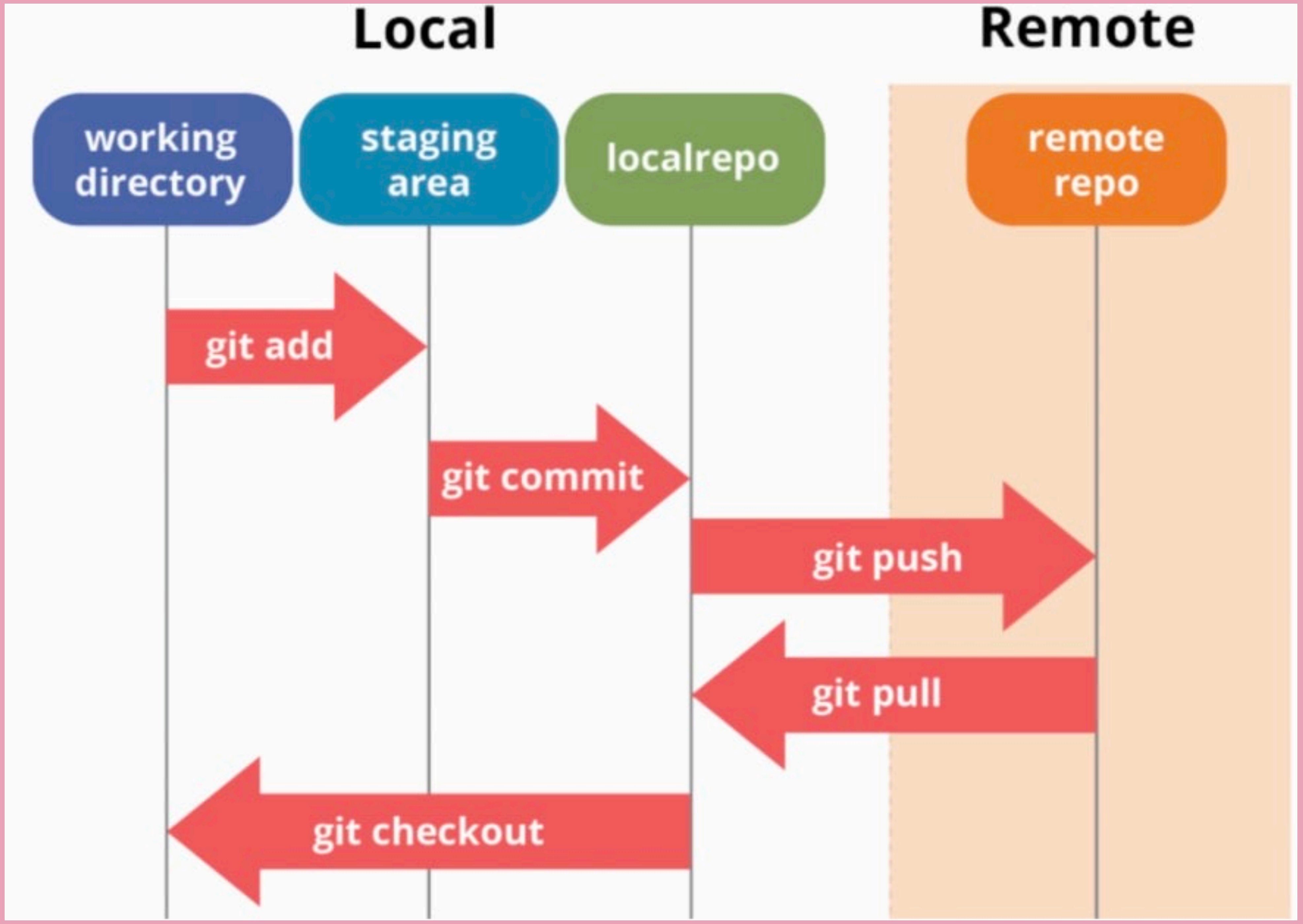
- Considerations for people in bioinformatics
  - Not suited for large sequence files
  - Or binary compiled programs
- Free GitHub accounts don’t have strict privacy protections
  - Make sure to sanitize any identifiable information from data
- Don’t save passwords (you would be surprised how many people do!!)

Simple Workflow

How do we collaborate?

Agree on a common workflow.

This workflow works well for small teams (1-3 people) working asynchronously and on separate parts of the project



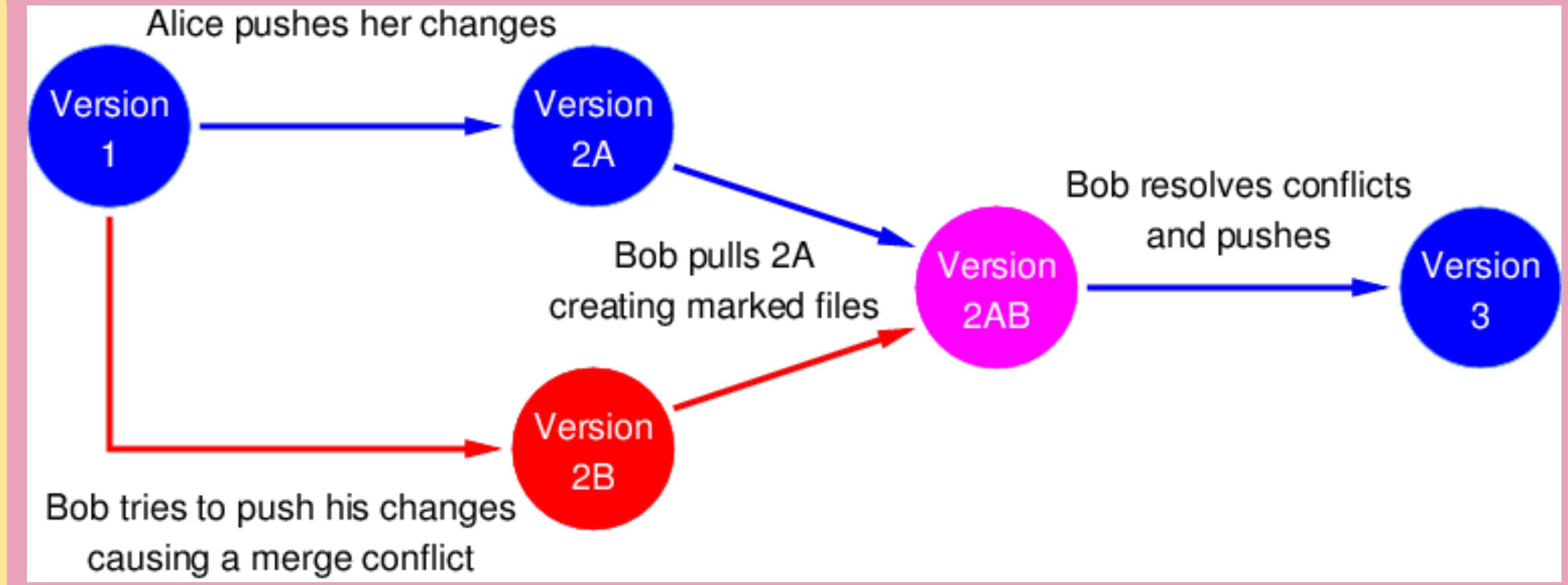
Downsides of Simple Workflow?

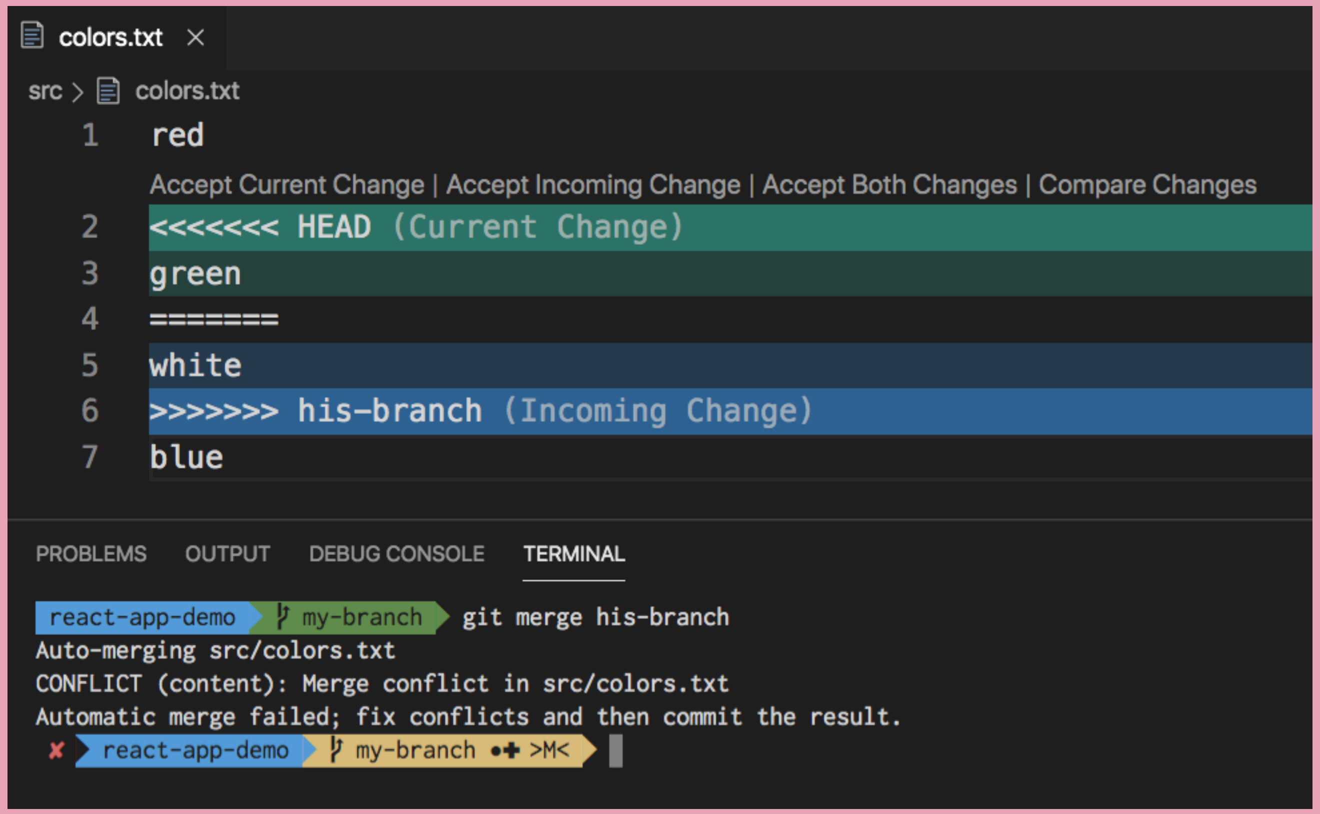
Merge Conflicts –

When editing same file

Why we pull before we push

Want all conflicts to take place locally!


Alice pushes her changes
Version 1
Version 2A
Bob pulls 2A creating marked files
Version 2AB
Bob resolves conflicts and pushes
Version 3
Version 2B
Bob tries to push his changes causing a merge conflict

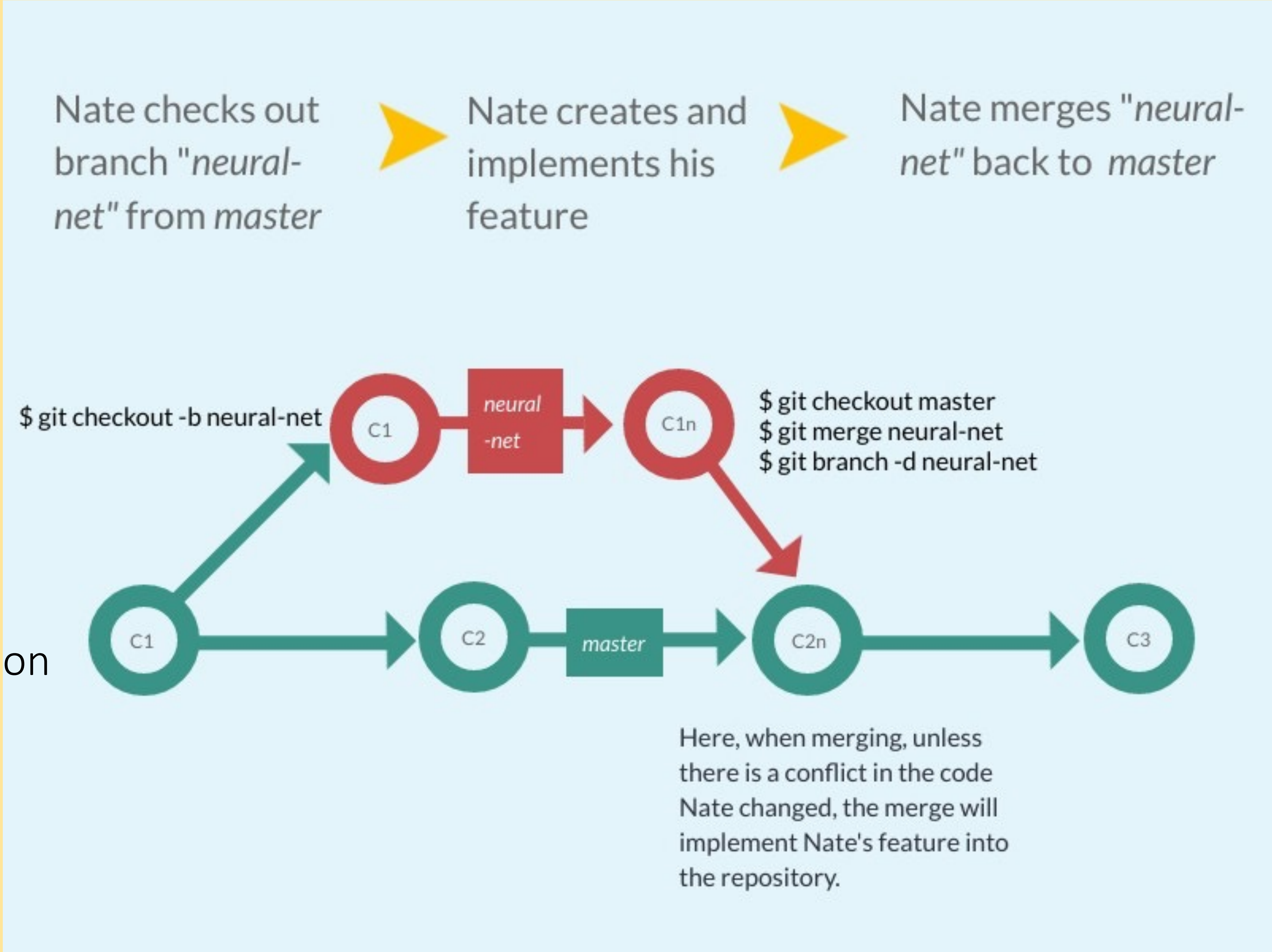

colors.txt
src > colors.txt
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL

```
red
<<<<<<< HEAD (Current Change)
green
=======
white
>>>>>>> his-branch (Incoming Change)
blue
```

```
react-app-demo  my-branch  git merge his-branch
Auto-merging src/colors.txt
CONFLICT (content): Merge conflict in src/colors.txt
Automatic merge failed; fix conflicts and then commit the result.
✘  react-app-demo  my-branch ●✚ >M<
```

Branch Workflow

- One step up
- Create a branch with your changes
- Once you are satisfied – have a discussion with your colleagues
- Then merge into the main branch



Nate checks out branch "*neural-net*" from *master*

Nate creates and implements his feature

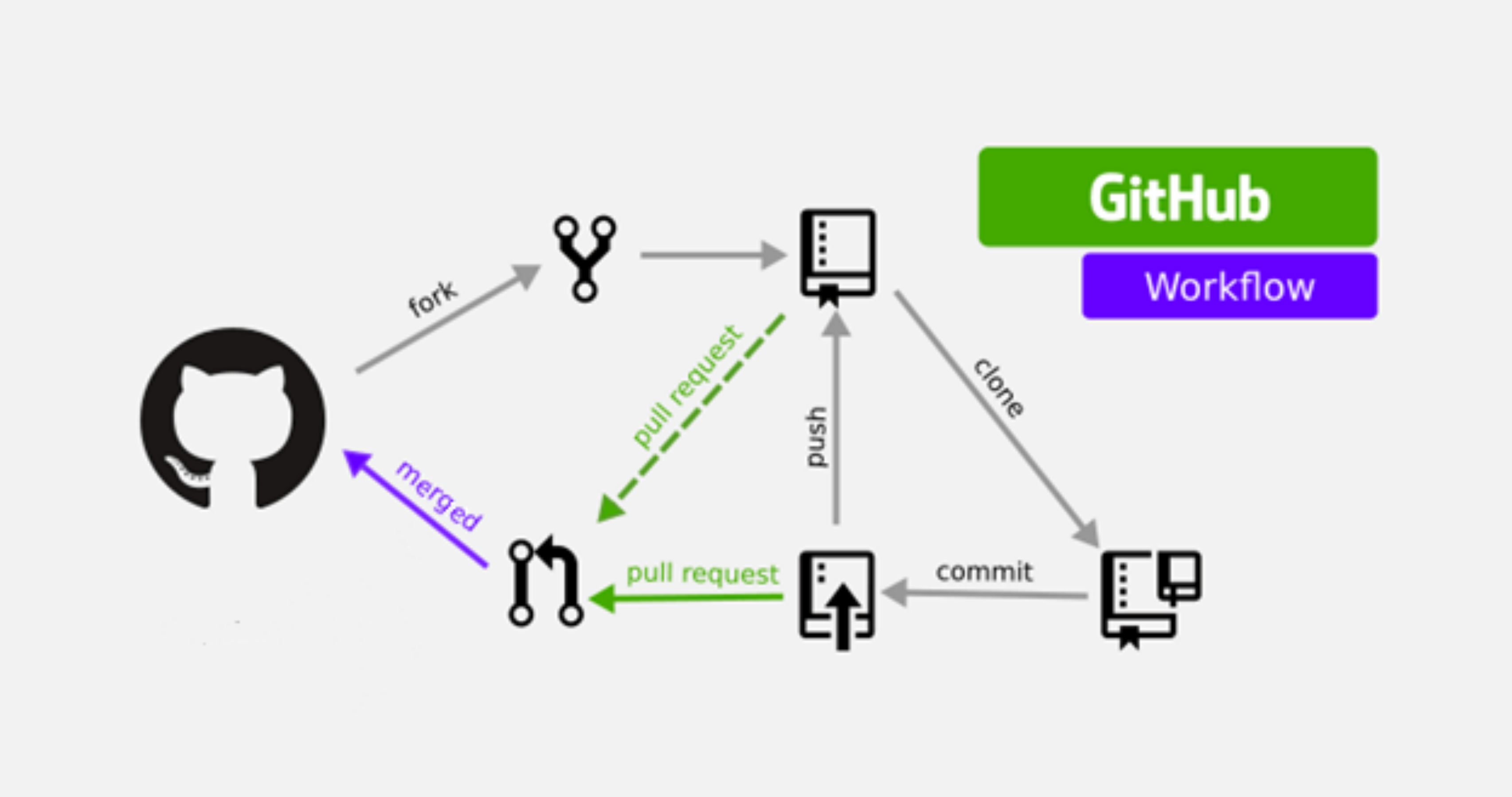
Nate merges "*neural-net*" back to *master*

```
$ git checkout -b neural-net
```

```
$ git checkout master
$ git merge neural-net
$ git branch -d neural-net
```

Here, when merging, unless there is a conflict in the code Nate changed, the merge will implement Nate's feature into the repository.

Fork workflow



- Good for very large projects
- Or ones where developers are not working closely together all the time (like a lot of bioinformatics projects!)
- For contributing to others' open source code – be a good citizen!

Helpful Tools

Overview Release Notes Help

GitHub Desktop

Focus on what matters instead of fighting with Git. Whether you're new to Git or a seasoned user, GitHub Desktop simplifies your development workflow.

Get The Legendary GitKraken Client

Includes an intuitive Git GUI & powerful Git CLI

Thank you!