Version control using Git

https://github.com/nih-fmrif/git-training

Session 2

12/12/2019

Data Science and Sharing Team
NIMH

Logistics

One more session:

2/19 from 1:30 - 3:30, same room (B1C208)

Today:

- Lecture - Review, going back in time in your local repo
- Walkthrough - Going back in time
- Lecture - Github, documentation, and remote changes
- Walkthrough - Fork, edit, pull
- Exercise

https://github.com/nih-fmrif/git-training

Review: repositories and .git

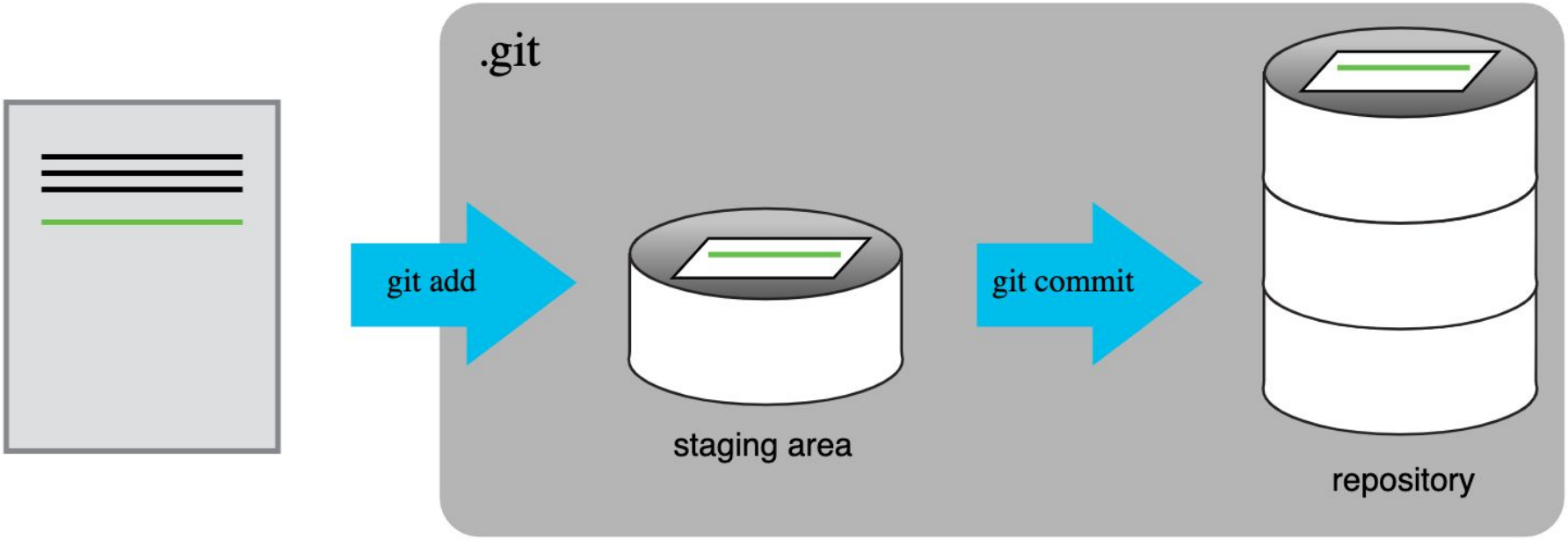


Image from: http://swcarpentry.github.io/git-novice/04-changes/index.html

Review: git workflow

- Change your files (edit or create new)

Image from: http://practicalseries.com/1002-vcs/02-02-concept.html

Review: git workflow

- Add changes to staging area

```
git add myNewFile.txt
```

Image from: http://practicalseries.com/1002-vcs/02-02-concept.html

Review: git workflow

- Commit the changes

```
git commit -m “adding
myNewFile.txt”
```

Image from: http://practicalseries.com/1002-vcs/02-02-concept.html

Review: git workflow

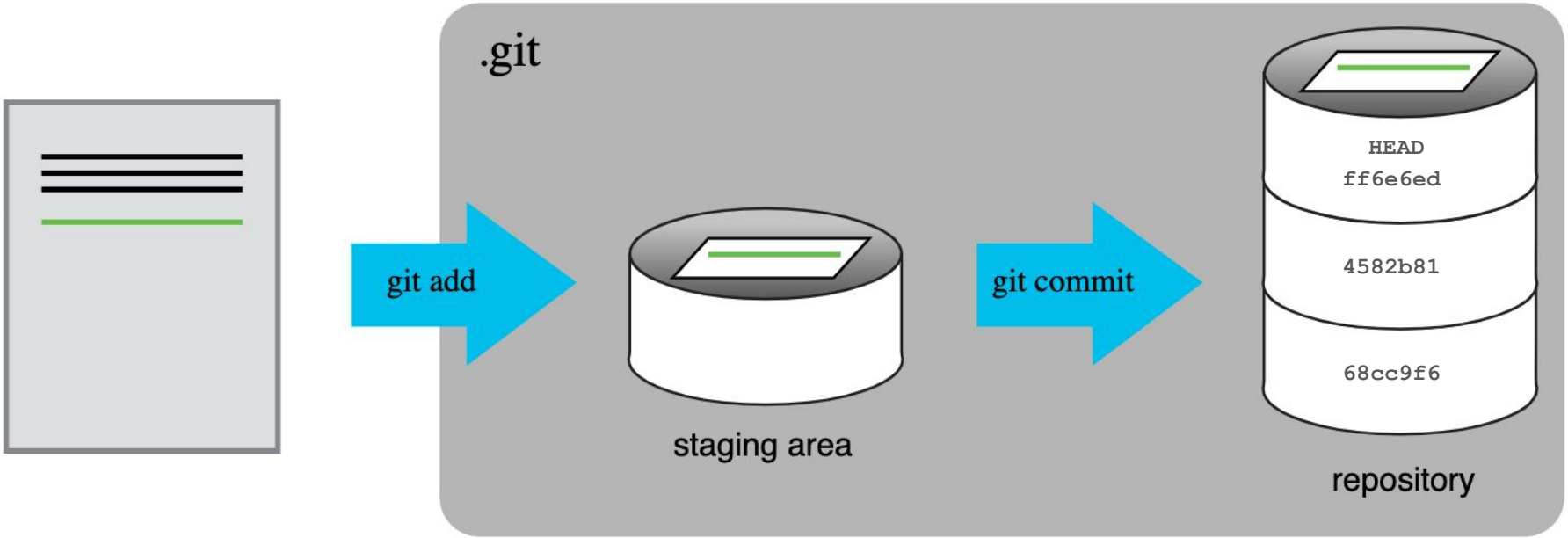
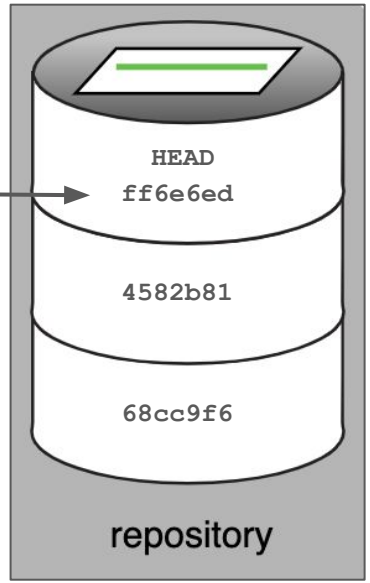


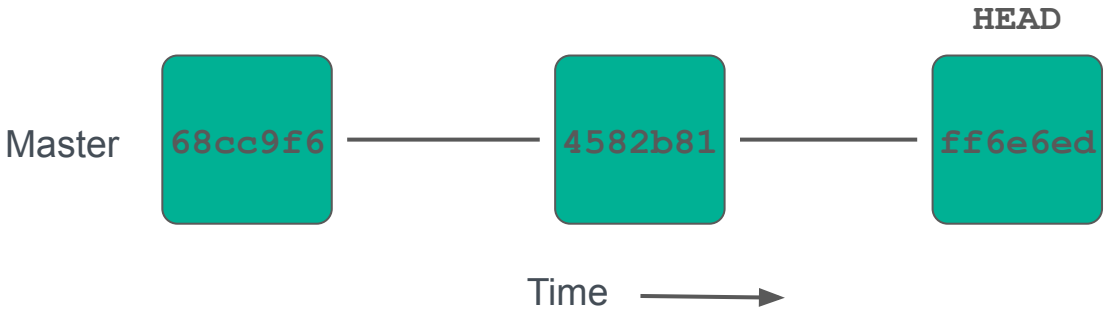
Image from: http://swcarpentry.github.io/git-novice/04-changes/index.html

Commit history

- *HEAD* - most recent commit
- Commit IDs
  - 40-character Secure Hash Algorithm 1 (SHA-1)
  - Typically referenced using first 7 characters

`ff6e6edf28e4947a6f38c3dfac5614e215725b52`





Git commands: checkout

Switch between different versions of a target entity

- **Files**, commits, or branches
- Changes the contents of the files and working directory to reflect target time point

---

```
git checkout <commit> <file>
```

Git commands: checkout

Switch between different versions of a target entity

- Files, **commits**, or branches
- Changes the contents of the files and working directory to reflect target time point

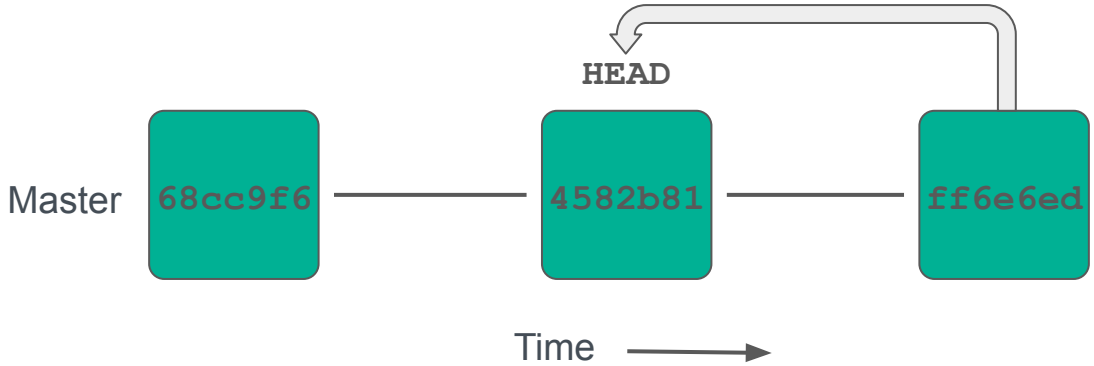
---

```
git checkout <commit>
```

Git commands: checkout

* You can `checkout` individual file histories or the entire repository

Git commands: checkout

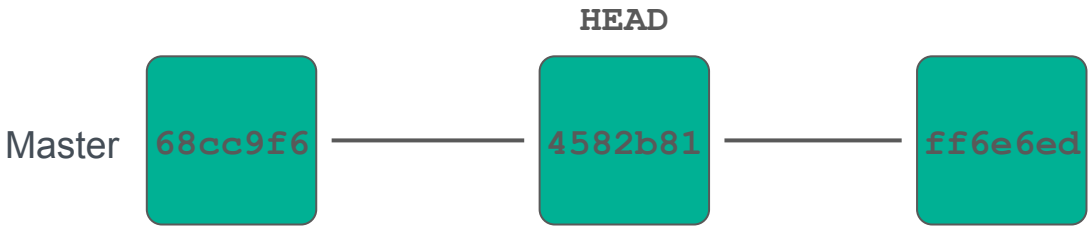



```
git checkout 4582b81
```

Git commands: checkout

Caution!

- `checkout` is useful for *inspecting*
- Changing files will put the HEAD in a detached state and you will lose changes if you commit them or change HEAD
- Solution: create a branch (more next week)



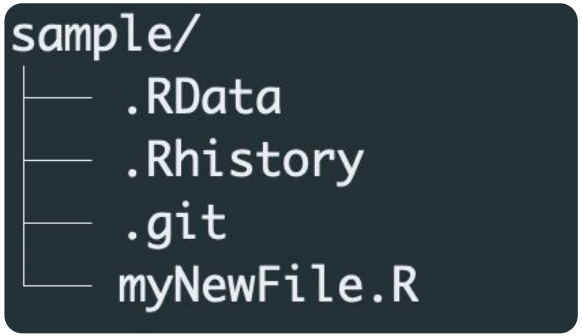
Where do changes go?

Git concept: ignoring files

What about files you do not want to track?

.gitignore

- Hidden text file in the root of the repository that contains filenames to be ignored from versioning
  - Hidden files created by OS
  - Temporary or intermediate files
  - Data
- Accepts wildcards and directories



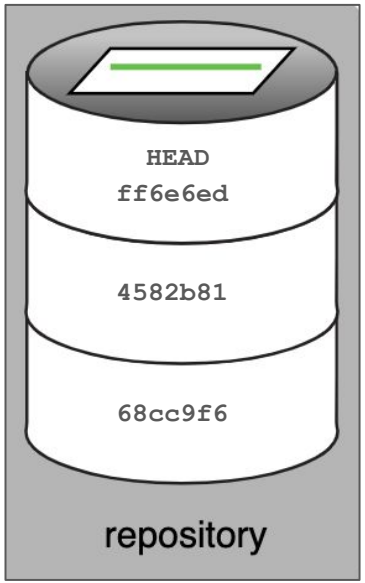
Example

Going back in time
and ignoring things

https://github.com/nih-fmrif/git-training/tree/master/session_2#example-1

Best practices: commits

- Commit often
- Commit related changes
- Do not commit half done work*
- Use informative commit messages
  - Agree on a workflow
  - Possible message tagging (eg *incremental build*)
- Test before you commit



https://www.git-tower.com/learn/git/ebook/en/command-line/appendix/best-practices

Review: git workflow

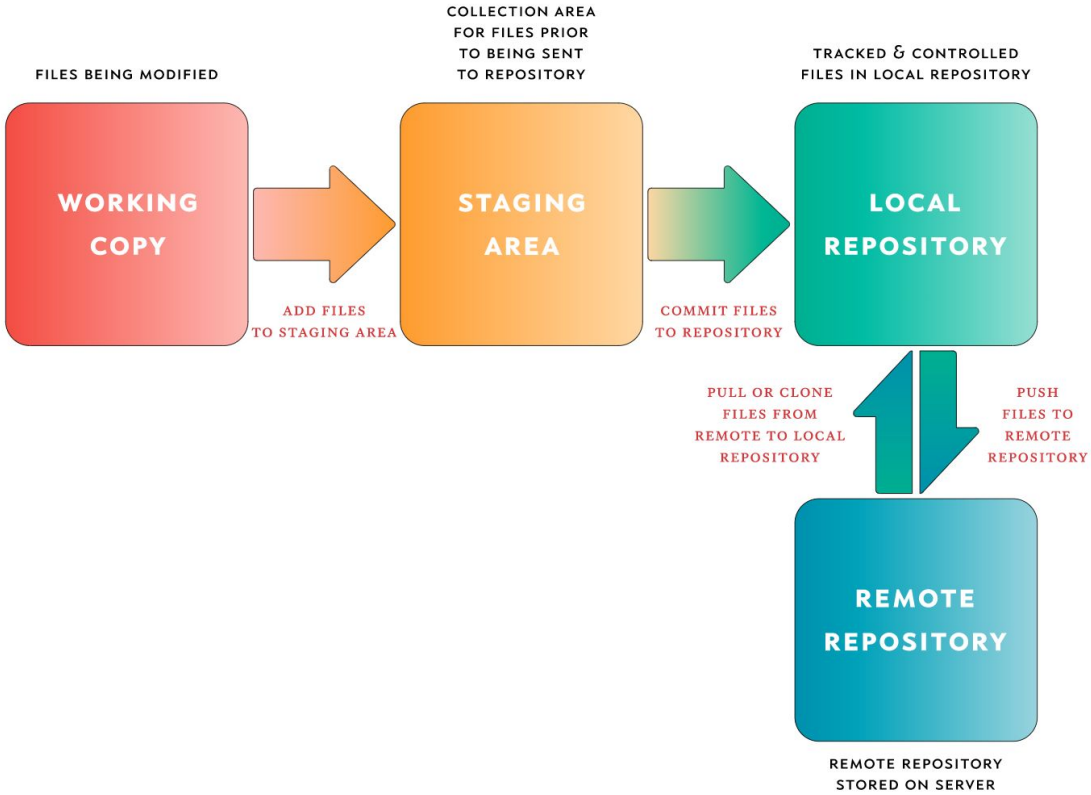
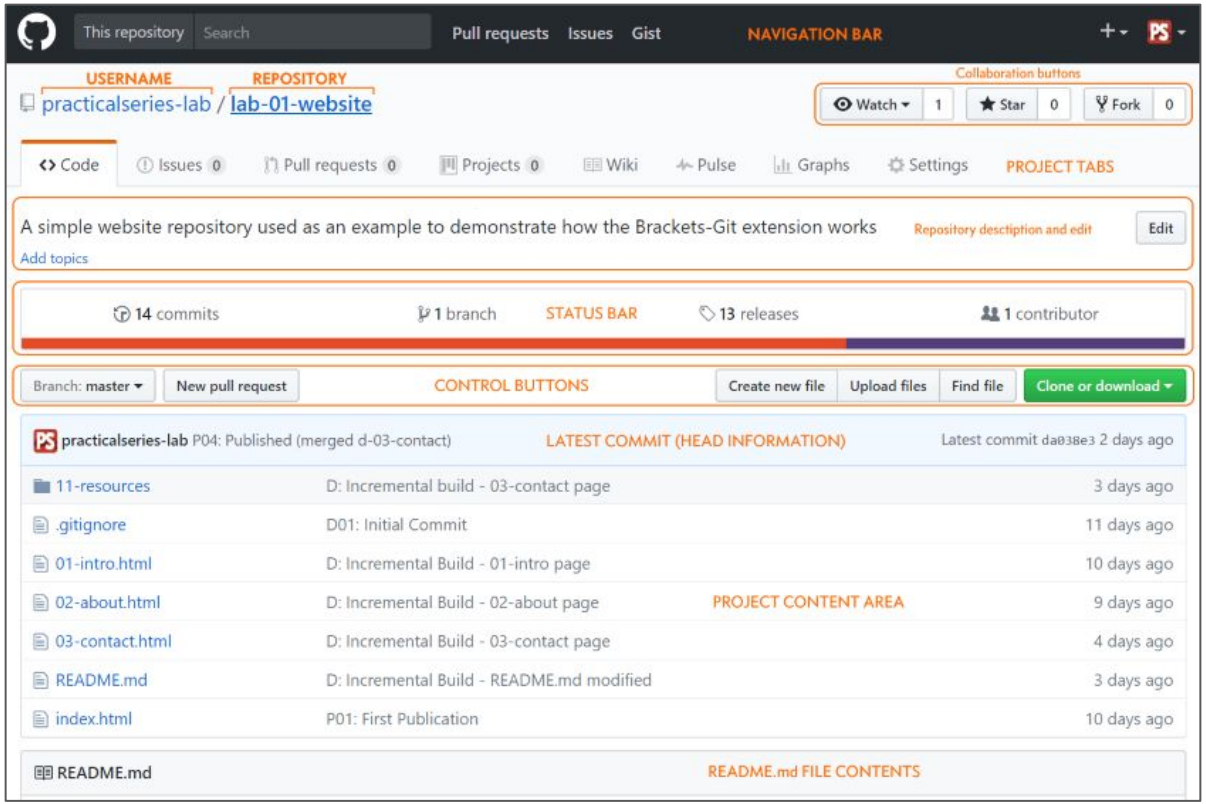


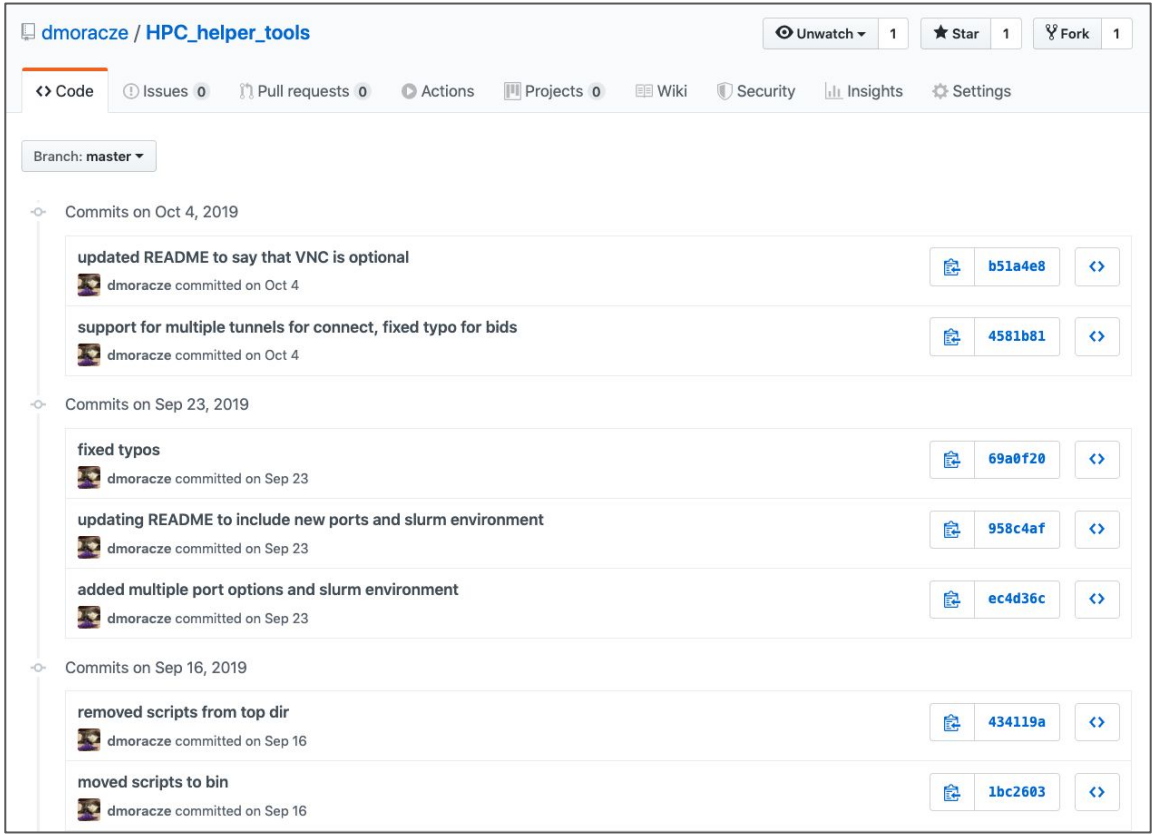
Image from: http://practicalseries.com/1002-vcs/02-02-concept.html

GitHub



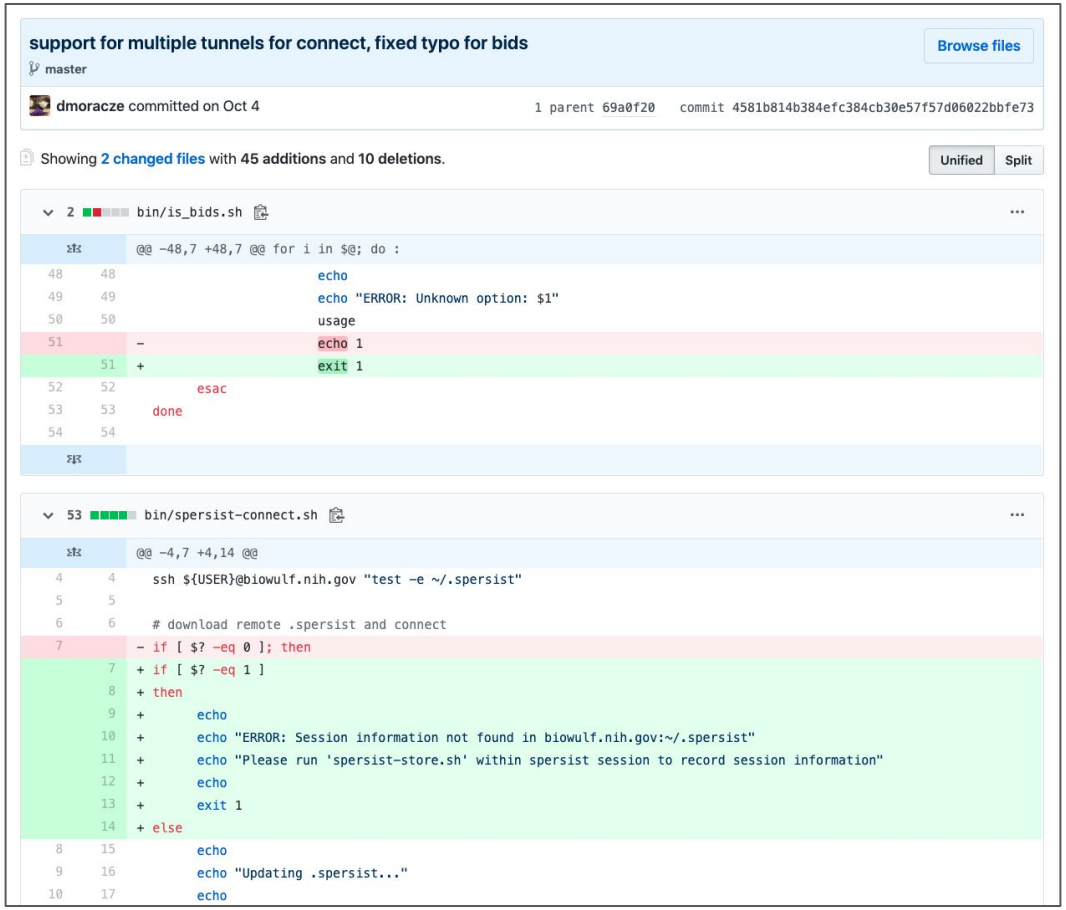
http://practicalseries.com/1002-vcs/09-01-github.html#js--090102

GitHub



https://github.com/dmoracze/HPC_helper_tools

GitHub



support for multiple tunnels for connect, fixed typo for bids

Browse files

master

dmoracze committed on Oct 4 — 1 parent 69a0f20 — commit 4581b814b384efc384cb30e57f57d06022bbfe73

Showing **2 changed files** with **45 additions** and **10 deletions**. Unified | Split

2 bin/is_bids.sh

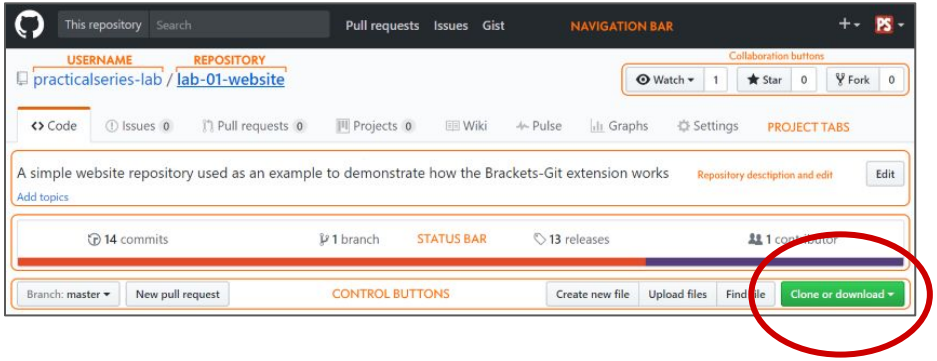
```
@@ -48,7 +48,7 @@ for i in $@; do :
48 48                       echo
49 49                       echo "ERROR: Unknown option: $1"
50 50                       usage
51    -                     echo 1
   51 +                     exit 1
52 52           esac
53 53   done
54 54
```

53 bin/spersist-connect.sh

```
@@ -4,7 +4,14 @@
4 4   ssh ${USER}@biowulf.nih.gov "test -e ~/.spersist"
5 5
6 6   # download remote .spersist and connect
7    - if [ $? -eq 0 ]; then
   7 + if [ $? -eq 1 ]
   8 + then
   9 +       echo
  10 +       echo "ERROR: Session information not found in biowulf.nih.gov:~/.spersist"
  11 +       echo "Please run 'spersist-store.sh' within spersist session to record session information"
  12 +       echo
  13 +       exit 1
  14 + else
8 15          echo
9 16          echo "Updating .spersist..."
10 17         echo
```

https://github.com/dmoracze/HPC_helper_tools

Git commands: clone

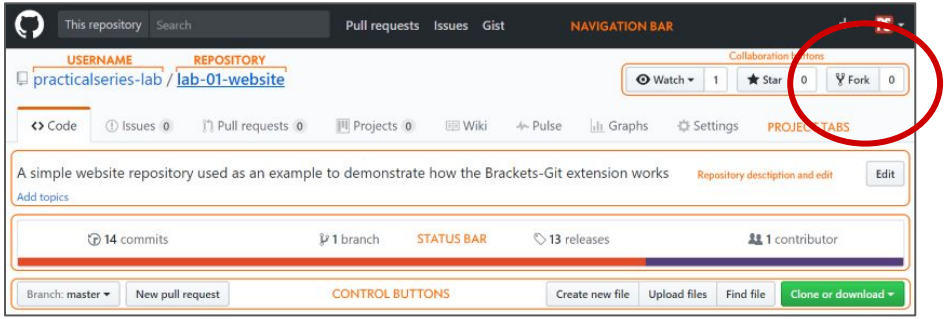


Download an existing remote repository to local

- Good practice to clone repo into a directory of the same name
- Changes will be pushed to the origin of the clone
- This should look familiar to last class's exercise

```
git clone <repo_url>
```

GitHub function: fork



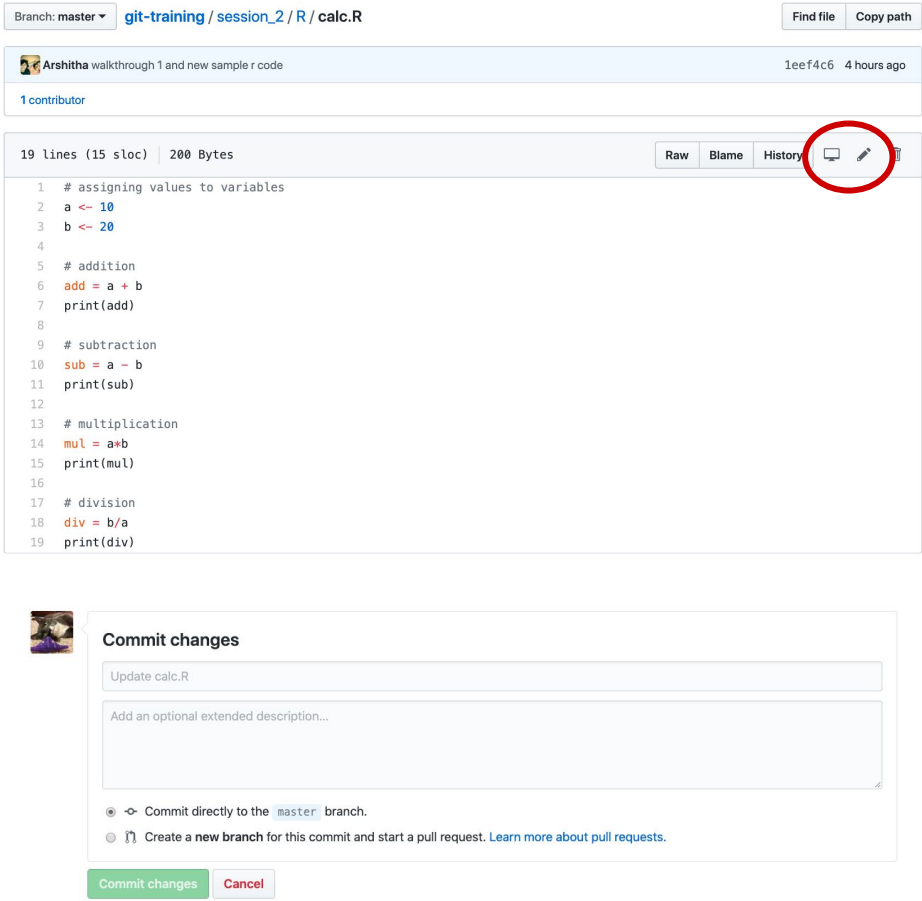
What if you want your own copy?

- Forking creates a personal copy of a repo, with a link back to the original
- You can then `clone` your personal copy to your local computer

Click ‘Fork’ button (no command line)

Editing files on GitHub

- All text files (eg, code, markdown, etc…) can be edited remotely
- To save changes, you'll need to add a commit
- What do you think happens to a local repo when a remote commit is made?



Git commands: pull

- Pull changes from a remote repository to your local copy
- Simplest example, one branch cloned from origin: `git pull`
- Could create conflicts (more next time)

`git pull <remote> <branch>`

Review: git workflow

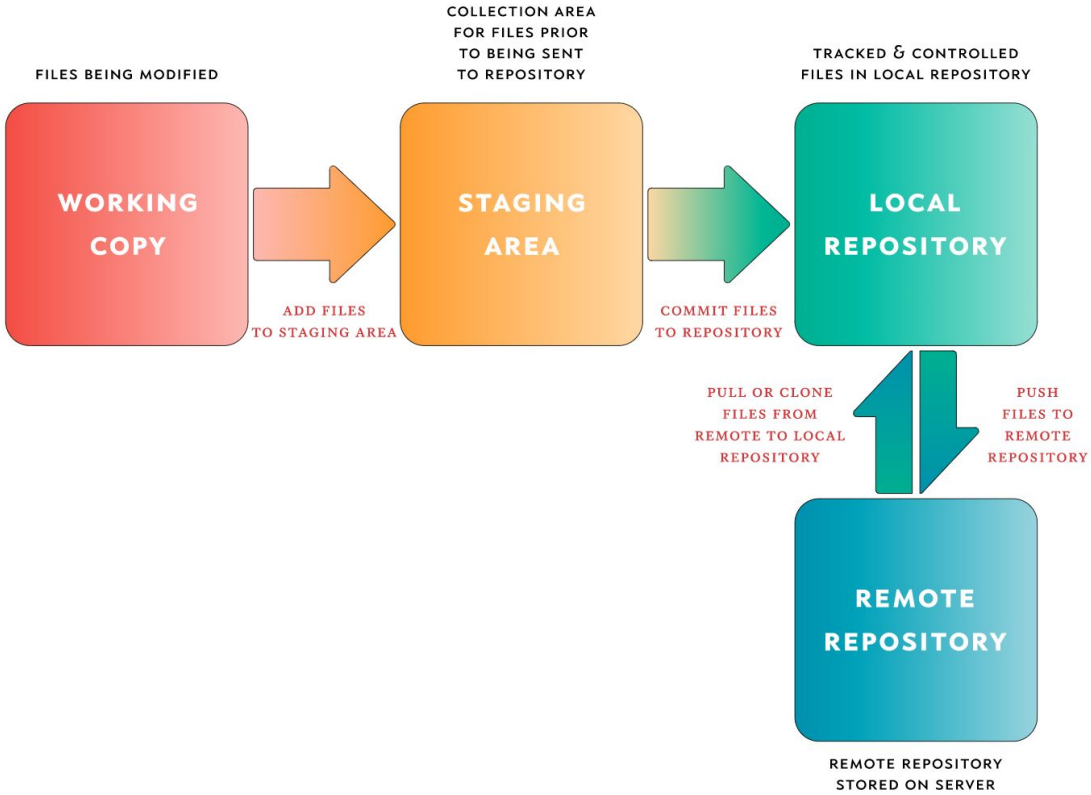
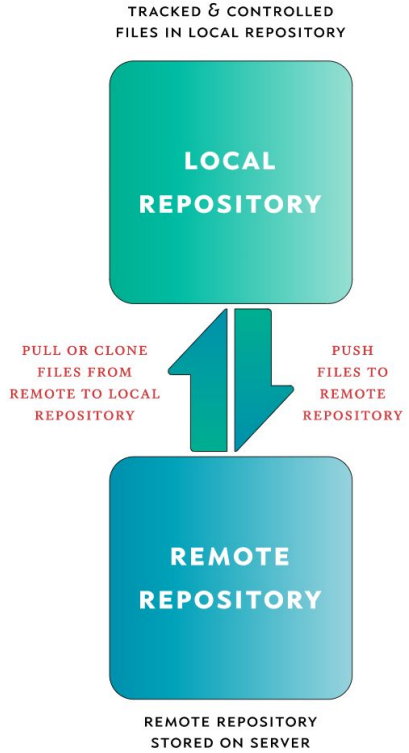


Image from: http://practicalseries.com/1002-vcs/02-02-concept.html

Best practices: remote

- Use the same same for local and remote repositories
- Update local repository frequently
  - Store local changes (`commit`)
  - Update local repo (`pull`)
- Update and resolve conflicts before pushing changes
  - Store local changes (`commit`)
  - Update local (`pull`)
  - Update remote (`push`)



http://practicalseries.com/1002-vcs/02-09-concept.html#js--020902

Documentation

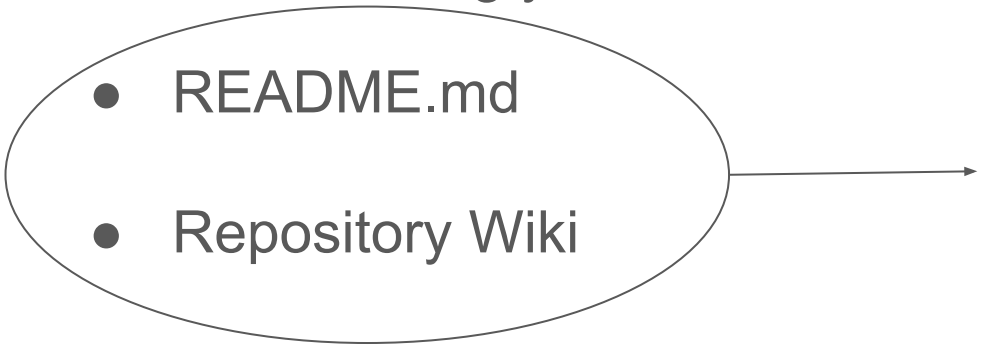
3 Levels

- Commenting your code (which you should be doing already, right?)
- README.md
- Repository Wiki

Documentation

3 Levels

- ~~Commenting your code~~ ~~(which you should be doing already, right?)~~
- README.md
- Repository Wiki



Both styles are repository-level documentation

Documentation: README.md

```
### creating local repository
```shell
# creating a directory on the desktop
cd ~/Desktop
mkdir sample
cd sample
ls -a
```
* Right now this is just another directory on my desktop. How do I make this a repo?

```shell
git init
ls -a
```

* .git sub-directory within the directory is what makes it a repository

creating a new file and tracking it with git
```shell
touch myNewFile.txt
echo 'Hello, world!' > myNewFile.txt

# checking which files are tracked and which aren't
git status

# adding a file to the staging area
git add myNewFile.txt

git status

# commiting the file
git commit -m "test file commit"

git status

vi myNewFile.txt
# add 2 new lines of text

# differences made after the last commit
git diff

git status

git add myNewFile.txt

git commit -m "adding two new lines"
```
```

creating local repository

```shell
# creating a directory on the desktop
cd ~/Desktop
mkdir sample
cd sample
ls -a
```

- Right now this is just another directory on my desktop. How do I make this a repo?

```shell
git init
ls -a
```

- .git sub-directory within the directory is what makes it a repository

creating a new file and tracking it with git

```shell
touch myNewFile.txt
echo 'Hello, world!' > myNewFile.txt

# checking which files are tracked and which aren't
git status

# adding a file to the staging area
git add myNewFile.txt

git status

# commiting the file
git commit -m "test file commit"

git status

vi myNewFile.txt
# add 2 new lines of text

# differences made after the last commit
git diff

git status

git add myNewFile.txt

git commit -m "adding two new lines"
```

Documentation: Wiki

bids-standard / bids-starter-kit

Watch 25 | Star 97 | Fork 49

Code | Issues 18 | Pull requests 2 | Actions | Projects 1 | Wiki | Security | Insights

Editing Home

Page History | New Page

Home

Write | Preview

h1 h2 h3 | B i | Edit mode: Markdown

```
<p align="center">
  <img src ="https://i.imgur.com/1Ox6N6R.png" />

<br>

</p>

<h3> Welcome to the home page of the wiki! </h3>

This is a community edited source of information and collection of links to help you get started with BIDS.
Some resources include a description of BIDS-compatible folder structures, a glossary to help with some of the
technical terms, and some tips and tricks to get you started.

The various pages in this wiki can be found by clicking "pages" at the top right side of this screen 👉

We also have a list of the most common places to start in the sidebar (also on the right 👉)

<br>

Please note that this wiki is actively maintained and still growing, so please help us add anything that you feel
would be useful! You can read more information about [how to contribute in our guidelines]
(https://github.com/INCF/BIDS-Starter-Kit/blob/master/CONTRIBUTING.md).
```

Edit message

Write a small message here explaining this change. (Optional)

Save Page

bids-standard / bids-starter-kit

Watch 25 | Star 97 | Fork 49

Code | Issues 18 | Pull requests 2 | Actions | Projects 1 | Wiki | Security | Insights

Home

Edit | New Page

Kirstie Whitaker edited this page on Oct 3, 2018 · 15 revisions

BIDS STARTER KIT WIKI

Simple tips and references for an easy start to BIDS

Welcome to the home page of the wiki!

This is a community edited source of information and collection of links to help you get started with BIDS. Some resources include a description of BIDS-compatible folder structures, a glossary to help with some of the technical terms, and some tips and tricks to get you started.

The various pages in this wiki can be found by clicking "pages" at the top right side of this screen 👉

We also have a list of the most common places to start in the sidebar (also on the right 👉)

Please note that this wiki is actively maintained and still growing, so please help us add anything that you feel would be useful! You can read more information about how to contribute in our guidelines.

Want to add to this wiki? Check out our contributing guidelines

For any questions, please contact us

Pages 16

Find a Page...

- Home
- BIDS Apps Help
- BIDS Extentions
- BIDS Validator info
- Community Resources
- Contact
- Creating a BIDS compatible iEEG dataset
- Dataset Examples
- FAQ
- Glossary
- Metadata file formats
- Publications
- The BIDS folder hierarchy
- Tutorials
- Tutorials FAQ
- Useful Links

Home

Frequently Asked Questions

Documentation: Which one?

README.md

- ‘Front page’ of project
- Markdown syntax
- Files stored in repo
- Editing requires changes to repo

Wiki

- In depth documentation
- Markdown syntax
- Files stored remotely
- Edits can be made freely

Really, it comes down to preference...

Example

Forking, cloning, remote, and local changes

https://github.com/nih-fmrif/git-training/tree/master/session_2#example-2

Exercise

Putting it all together

https://github.com/nih-fmrif/git-training/tree/master/session_2#exercise