

GIT



Creating a Repository in Gitlab

Some useful notation

Git init - Creates a repository in the folder that the bash window is currently open in

Git add - Adds files, either by specific name or by using the period symbol . to say ‘i am adding every file that has been changed’. This can be dangerous in large projects!!

Git push - Pushes your changes to your remote repository, think of it as updating any old files with your new changes

Git pull - Downloads and merges your files whereas

Git fetch - Downloads changed files without any merging

Git commit - Creates a snapshot of your current repository in case of any problems that need you to revert to an old status of your project

Git stash - Saves / shelves your changes as if to hold them for later if needed.

Git stage - Adds files to the staging area, ready for the next commit

Git diff - Shows the changes between the commits and the working tree

Git checkout - Switch branches that you are working on

Usual flow of a project

Git init - This is if you have code you would like to upload

OR

Git clone <your link> - If you are trying to download a repository

THEN

Git remote add origin <your link> - Connecting the remote repository to your local repository

Git add . - Adding every file

Git commit -m “Your Message” - Committing your changes to be pushed

Git push -u origin master - Push your changes to the origin repository in the master branch

Then to update your code you call

Git pull / git fetch - Merges / downloads your current code and the most recent code from the repository

And to update the repository with your new code you would start back at Git add . and the loop continues



General GitLab layout

gitlab.eecs.wsu.edu/44436

Apps YouTube Gmail Blackboard Google Calendar OSBLE

GitLab Projects Groups Activity Milestones Snippets

New project
New group
New snippet

nwisner
@44436 · Member since January 15, 2019

Overview Activity Groups Contributed projects Personal projects Starred projects Snippets

Sep Oct Nov Dec Jan Feb Mar Apr May Jun Jul Aug

M W F

Issues, merge requests, pushes, and comments.

Activity View all

- o nwisner @44436 Pushed new branch `part1` at `nwisner / SMILE Repo` 3 days ago
- o nwisner @44436 Pushed new branch `master` at `nwisner / SMILE Repo` 3 days ago
- o nwisner @44436 Pushed to branch `HW-1` at `nwisner / 321 HW` `a8127912` · Removed test code from friday class 3 days ago
- o nwisner @44436 Pushed to branch `HW-1` at `nwisner / 321 HW` `2031e71b` · initial commit of `hw-1` 3 days ago

Personal projects View all

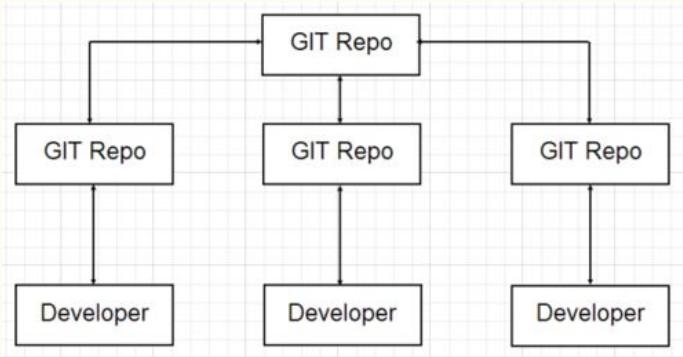
- 3 321 HW Maintainer Updated 3 days ago
- SMILE Repo Maintainer Updated 3 days ago

Git vs SVN (Subversion)

Git

Git stores the entire repository and changes onto your local machine

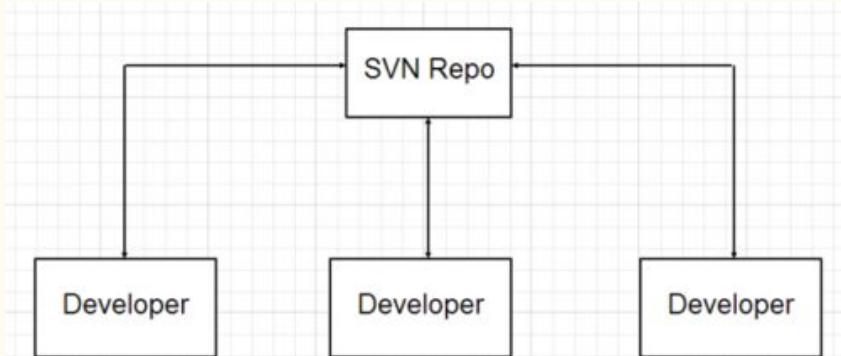
Git commits are local and the pushing is when it has remote updates



SVN

SVN lets users checkout the entire repository or just a branch

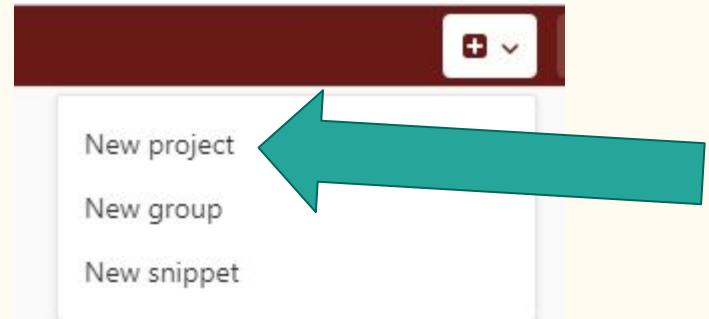
Since the SVN is stored remotely, our commits are not local but in the central server



Click the plus

Then choose New Project for a new repository

We will not cover the others today



Now this is your screen

Here is your repository name

Here is a description area

Privacy

Readme

New project

A project is where you house your files (repository), plan your work (issues), and publish your documentation (wiki), among other things.

All features are enabled for blank projects: from templates, or when importing, but you can disable them afterward in the project settings.

Information about additional Pages templates and how to install them can be found in our Pages getting started guide.

Tip: You can also create a project from the command line. Show command

Blank project Create from template Import project

Project name: sample-repo

Project URL: https://gitlab.eecs.wsu.edu/44436/ Project slug: sample-repo

Want to house several dependent projects under the same namespace? [Create a group](#).

Project description (optional)

Description format:

Visibility Level:

- Private
Project access must be granted explicitly to each user.
- Internal
The project can be accessed by any logged in user.
- Public
The project can be accessed without any authentication.

Initialize repository with a README
Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.

Create project Cancel

Always have your HW repositories be private!!

After creation with no readme

The screenshot shows a GitLab project named "sample-repo". The sidebar on the left lists various project management sections like Issues, Merge Requests, CI / CD, Operations, Wiki, Snippets, and Settings. A red arrow points from the "Settings" section in the sidebar to the "Command line instructions" section on the main page.

sample-repo Project ID: 1973

Add license

The repository for this project is empty

You can create files directly in GitLab using one of the following options.

New file **Add README** **Add CHANGELOG** **Add CONTRIBUTING**

Command line instructions

You can also upload existing files from your computer using the instructions below.

Git global setup

```
git config --global user.name "nwisner"  
git config --global user.email "nwisner@eecs.wsu.edu"
```

Create a new repository

```
git clone git@gitlab.eecs.wsu.edu:44436/sample-repo.git  
cd sample-repo  
touch README.md  
git add README.md  
git commit -m "add README"  
git push -u origin master
```

Push an existing folder

```
cd existing_folder  
git init  
git remote add origin git@gitlab.eecs.wsu.edu:44436/sample-repo.git  
git add .  
git commit -m "Initial commit"  
git push -u origin master
```

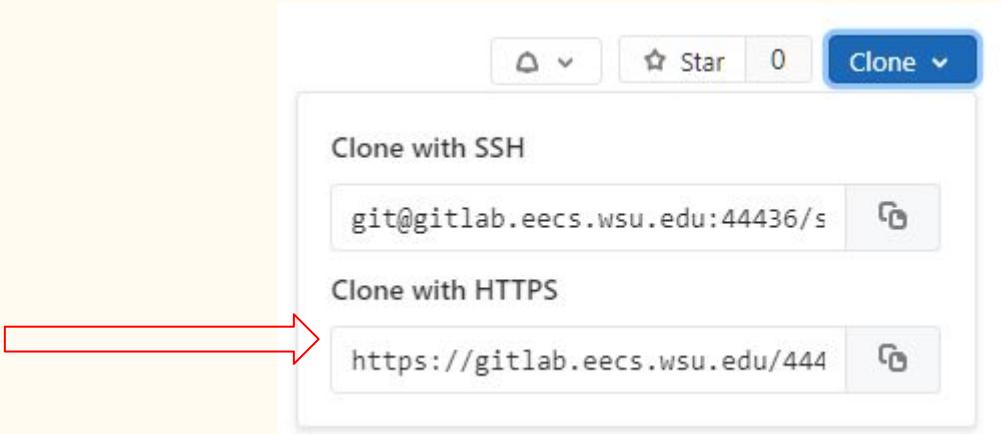
Push an existing Git repository

```
cd existing_repo  
git remote rename origin old-origin
```

Here are the instructions we will cover in detail

Your github link / URL

I recommend using
HTTPS, that is what we
will use today



We'll want to add our Gitlab link to connect our repositories

```
wisne@NWDeLL MINGW64 ~/Documents/Programming/Projects/Git Demo (master)
$ git remote add origin https://github.com/Nathan-Wisner/testing-Repo.git
```

Think of this as adding a connection called **origin** where pushing and pulling will get or give the latest code

I created a simple document to add

 a simple document to commit

8/26/2019 9:48 PM

Microsoft Word Doc...

0 KB

Think of this as your code, this can be one or one hundred files that can be added

I then used “git add .”

And yes the . is important

Git add . will add every file *that has been changed* inside that folder to your repository.

This could be your one file or one hundred new ones.

If you have 100 files but only edited one, *only one file will be added*

```
wisne@NWDeLL MINGW64 ~/Documents/Programming/Projects/Git Demo (master)
$ git add .
```

```
wisne@NWDeLL MINGW64 ~/Documents/Programming/Projects/Git Demo (master)
$ |
```

Now we can commit this

Committing is basically like saving a copy of your project / repository.

It's as if you take a snapshot of how everything is and it can be uploaded or at least stored in case it is needed to be reverted in instance of a problem

```
wisne@NwDell MINGW64 ~/Documents/Programming/Projects/Git Demo (master)
$ git commit -m "First commit"
[master (root-commit) 5a37cae] First commit
1 file changed 0 insertions(+), 0 deletions(-)
create mode 100644 a simple document to commit.docx
```

Git commit messages

Our messages should always be about the changes made in that commit.

For instance, if we add a function to a file, the commit message should say “Added function x, function x does z”.

This is your way of communicating what was done to inform others and trace bugs and errors quickly

And finally push it to Gitlab

Pushing is simply uploading the changes to your remote repository called origin and the master branch

```
wisne@NWDe11 MINGW64 ~/Documents/Programming/Projects/Git Demo (master)
$ git push origin master
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 604 B | 302.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0)
To https://github.com/Jathan-Wisner/test-Git-Repo.git
    21a2ccf..6aa5017 master -> master
```

Think of it as git push *remote repository name* *branch name*

Now we can go to Gitlab and see the changes

Our changes

 nwisner	Merge branch 'master' of https://github.com/Nathan-Wisner/testing-Repo	...
 README.md		
 a simple document to commit.docx		

Our commit message

Initial commit
First commit

Time since the file was last updated

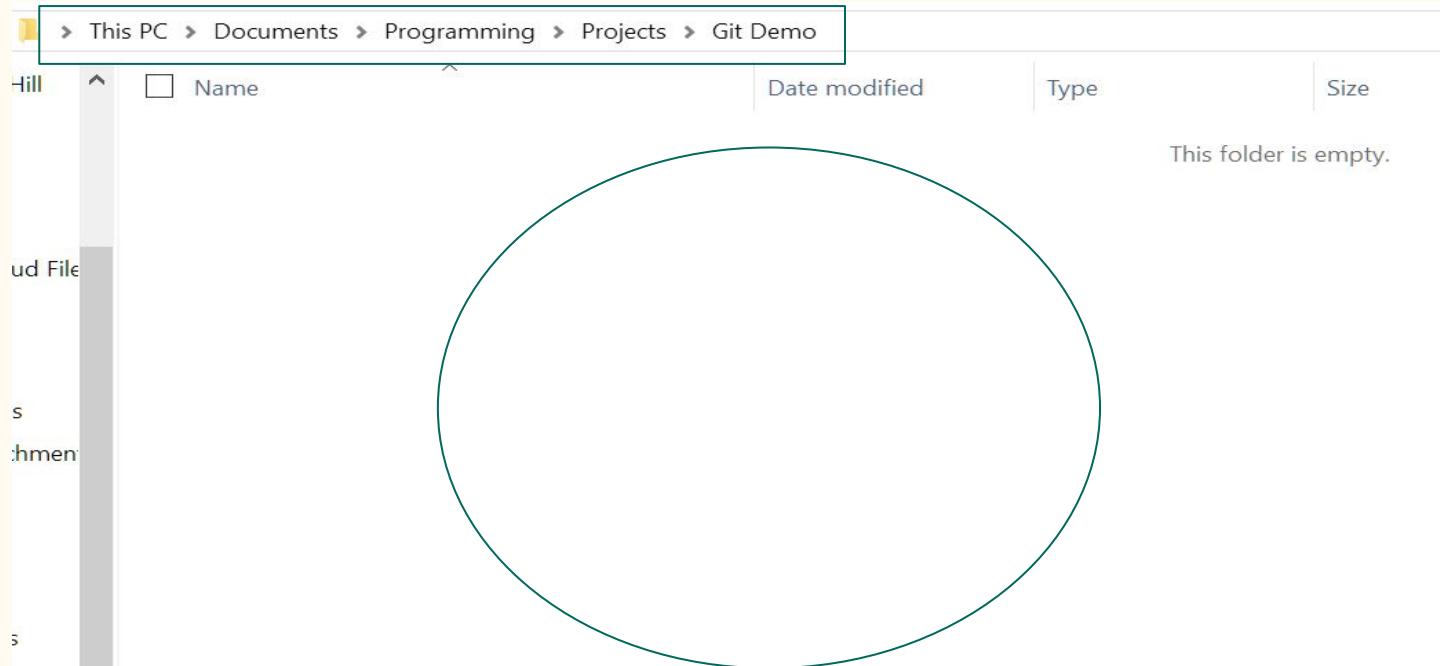
Latest commit 6aa50	3 minutes ago
	1 hour ago

10 minutes ago

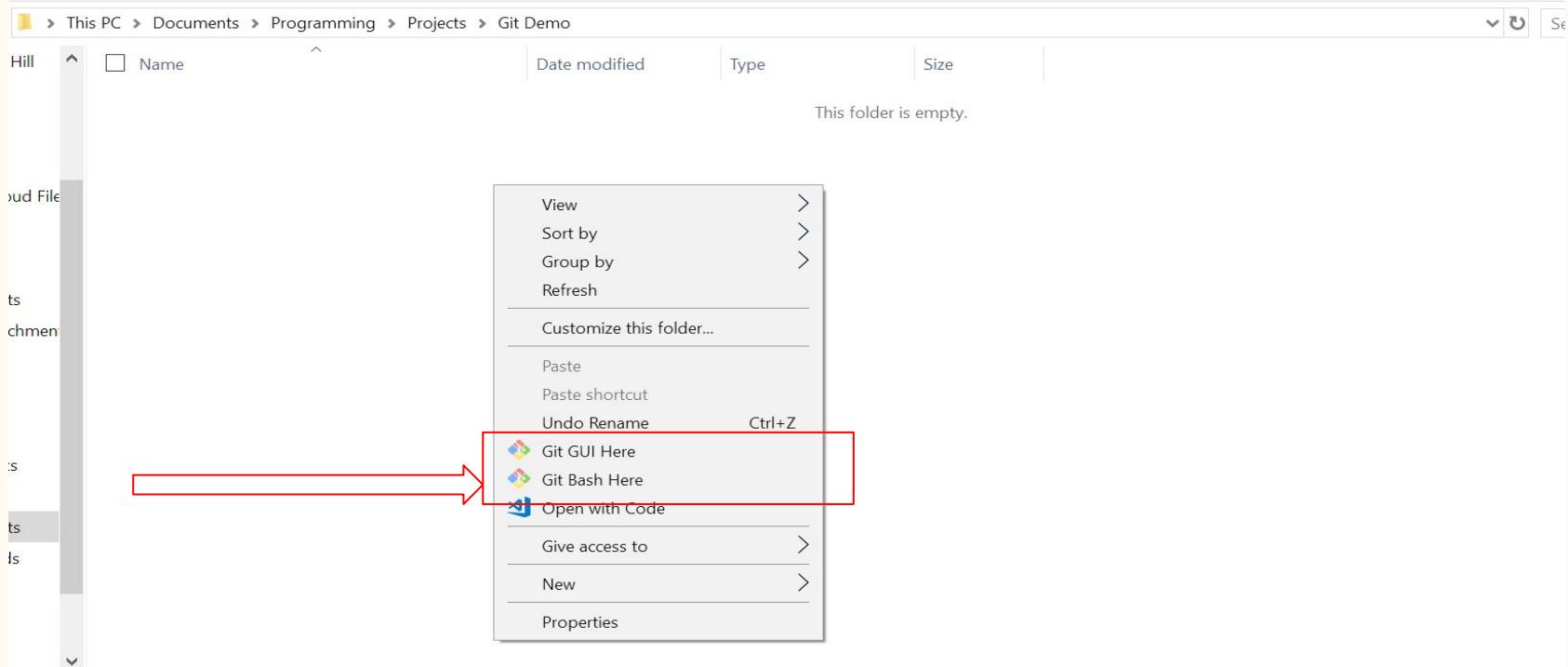
This example was started with a README , so ignore the README

Creating a repository in Git
Bash - For individual
review

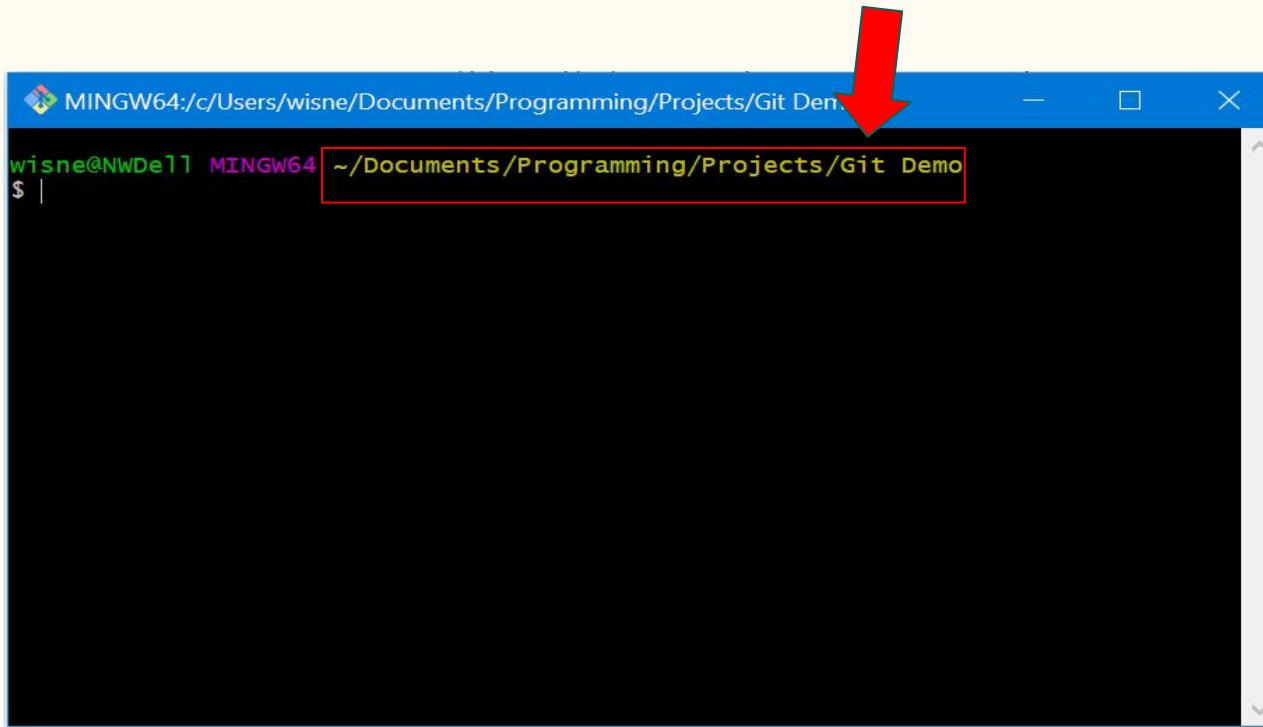
Click anywhere inside your project folder



Click “Git Bash Here”

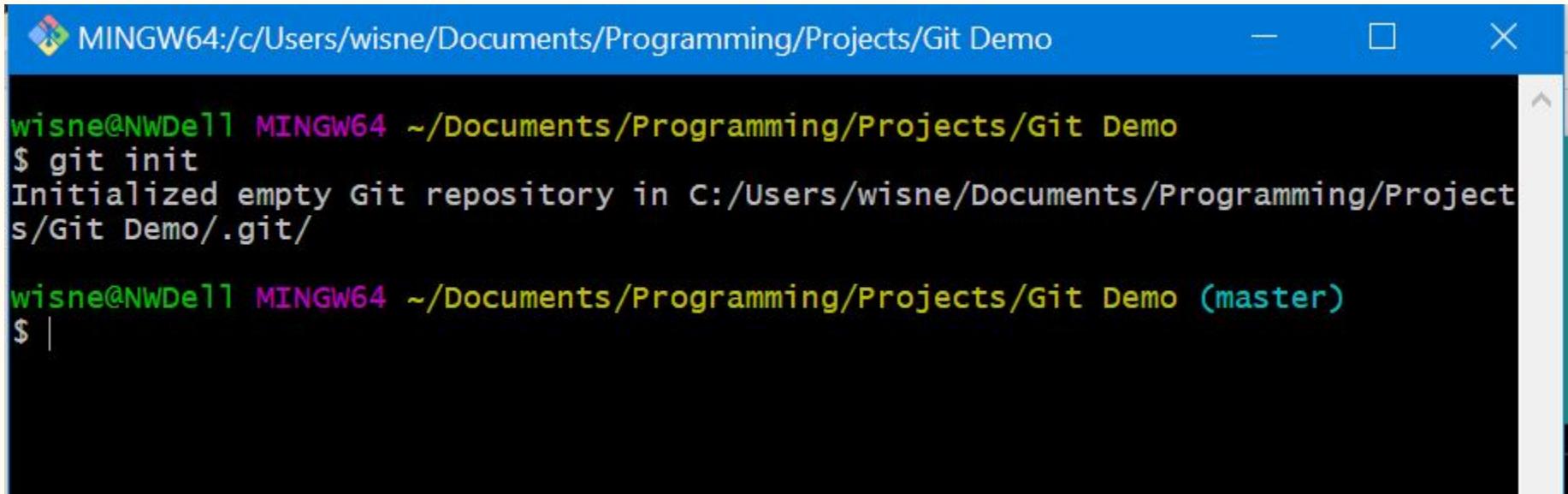


This is what is created, note the path here



Typing git init creates this repository locally on your machine

This is created in the area that you right clicked so keep that in mind!

A screenshot of a Windows terminal window titled "MINGW64:/c/Users/wisne/Documents/Programming/Projects/Git Demo". The window shows a command-line session where the user runs "git init" in their current directory. The output indicates that an empty Git repository was initialized in the specified location. The terminal has a blue header bar and a black body with white text.

```
wisne@NWDell MINGW64 ~/Documents/Programming/Projects/Git Demo
$ git init
Initialized empty Git repository in C:/Users/wisne/Documents/Programming/Projects/Git Demo/.git/
wisne@NWDell MINGW64 ~/Documents/Programming/Projects/Git Demo (master)
$ |
```

We'll want to add our Github link to connect our repositories

```
wisne@NWDeLL MINGW64 ~/Documents/Programming/Projects/Git Demo (master)
$ git remote add origin https://github.com/Nathan-Wisner/testing-Repo.git
```

Think of this as adding a connection called **origin** where pushing and pulling will get or give the latest code

I created a simple document to add

 a simple document to commit

8/26/2019 9:48 PM

Microsoft Word Doc...

0 KB

Think of this as your code, this can be one or one hundred files that can be added

I then used “git add .”

And yes the . is important

Git add . will add every file *that has been changed* inside that folder to your repository.

This could be your one file or one hundred new ones.

If you have 100 files but only edited one, *only one file will be added*

```
wisne@NWDeLL MINGW64 ~/Documents/Programming/Projects/Git Demo (master)
$ git add .
```

```
wisne@NWDeLL MINGW64 ~/Documents/Programming/Projects/Git Demo (master)
$ |
```

Now we can commit this

Committing is basically like saving a copy of your project / repository.

It's as if you take a snapshot of how everything is and it can be uploaded or at least stored in case it is needed to be reverted in instance of a problem

```
wisne@NwDell MINGW64 ~/Documents/Programming/Projects/Git Demo (master)
$ git commit -m "First commit"
[master (root-commit) 5a37cae] First commit
1 file changed 0 insertions(+), 0 deletions(-)
create mode 100644 a simple document to commit.docx
```

Git commit messages

Our messages should always be about the changes made in that commit.

For instance, if we add a function to a file, the commit message should say “Added function x to file y, function x does z”.

This is your way of communicating what was done to inform others and trace bugs and errors quickly

```
Added addHikes in SplashActivity to move hikes to FireStore
```

And finally push it to Gitlab

Pushing is simply uploading the changes to your remote repository called origin and the master branch

```
wisne@NWDe11 MINGW64 ~/Documents/Programming/Projects/Git Demo (master)
$ git push origin master
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 604 B | 302.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0)
To https://github.com/Jathan-Wisner/test-Git-Repo.git
    21a2ccf..6aa5017 master -> master
```

Think of it as git push *remote repository name* *branch name*

Now we can go to Gitlab and see the changes

Our changes

 nwisner	Merge branch 'master' of https://github.com/Nathan-Wisner/testing-Repo	...
 README.md		
 a simple document to commit.docx		

Our commit message

Initial commit
First commit

Time since the file was last updated

Latest commit 6aa50	3 minutes ago
	1 hour ago

10 minutes ago

This example was started with a README , so ignore the README

Git log

This is the result of calling “Git log” on a repository with 2 commits

We have our head hash name in pink, general notes in red and the commit message in green

```
commit da79e53294d755492a12d407ff791c05b6e7609e
```

```
Author: Nathan Wisner <wisner.nathan@gmail.com>
```

```
Date: Wed Jun 12 20:39:23 2019 -0700
```

```
Update hiking_detail.xml
```

```
commit e0bdddb688c73e1ac98d395ca3589f9d2bd439ce
```

```
Author: Nathan Wisner <wisner.nathan@gmail.com>
```

```
Date: Wed Jun 12 20:38:34 2019 -0700
```

```
    Added addHikes in SplashActivity to move hikes to FireStore
```

```
commit ebcae839105aa0e0bd2f0da6983acf9a2a835e55
```

```
Author: Nathan Wisner <wisner.nathan@gmail.com>
```

```
Date: Mon Jun 10 07:16:49 2019 -0700
```

Git log notes

Commits have the most recent commit at the top

This can be reversed if desired with “Git log --reverse”

With this they will display with the first commit at the top

Commits

What is a Commit?

- A “snapshot” of your current state
 - Add, commit, push
 - Use helpful commit messages!
-

```
madeline@Madeline-MacBookAir ~/my-smile-app/static $ git status
On branch part2
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   create.html
        modified:   index.html
        modified:   styles/main.css
```

```
madeline@Madeline-MacBookAir ~/my-smile-app/static $ git diff create.html
diff --git a/static/create.html b/static/create.html
index 0f7fd0f..6e41924 100644
--- a/static/create.html
+++ b/static/create.html
@@ -29,11 +29,10 @@
        <select class="happiness-level-input" name="happiness-level">
            <option value="1">Happy</option>
            <option value="2">Really happy</option>
-           <option value="3">so estatic, I can't stop smiling!</option>
+           <option value="3">so exstatic, I can't stop smiling!</option>
        </select>
        <input type="submit" value="Post" class="submit-input my-button" />
-       <input class="cancel-button my-button" value='Cancel'>
-       <!-- FINISH ME: add the rest of the form fields -->
+       <button class='my-button'>Cancel</button>
    </form>
  </div>
</section>
madeline@Madeline-MacBookAir ~/my-smile-app/static $
```

```
madeline@Madeline-MacBookAir ~/my-smile-app/static $ git add create.html
madeline@Madeline-MacBookAir ~/my-smile-app/static $ git status
On branch part2
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   create.html

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

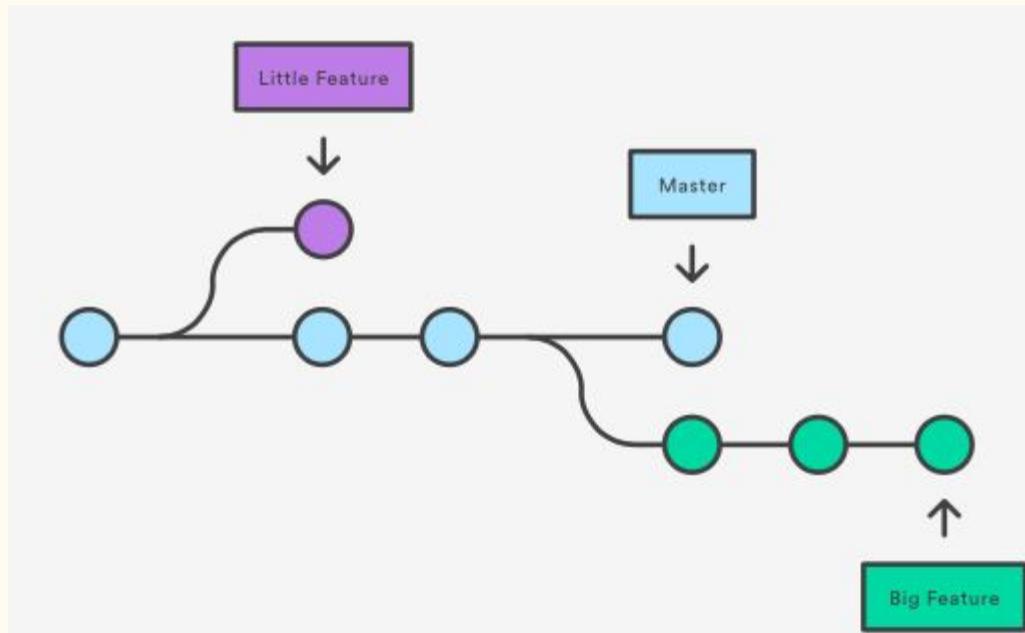
    modified:   index.html
    modified:   styles/main.css
```

```
madeline@Madeline-MacBookAir ~/my-smile-app/static $ git commit -m "Added compatibility for different browsers"
[part2 5eb8bc4] Added compatibility for different browsers
 1 file changed, 2 insertions(+), 3 deletions(-)
madeline@Madeline-MacBookAir ~/my-smile-app/static $ git push
```

Branching,
Merging,
Tagging,

Branching

- A pointer to a ‘snapshot’ of your changes
- When you want to work on a specific task, you branch
- Checkout, Branch, then Checkout
- Updates files in working directory



```
git branch <branch>
```

Create a new branch

```
git checkout <branchname>
```

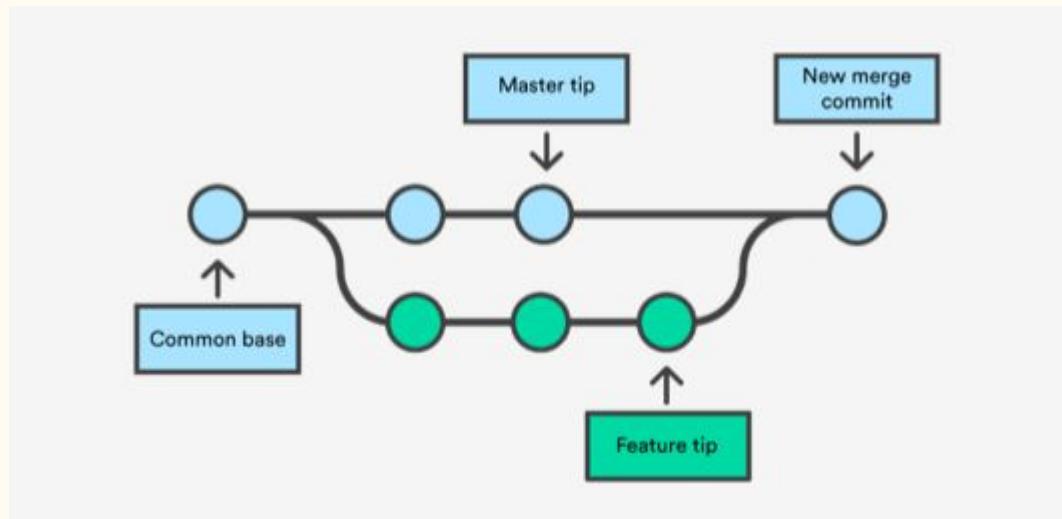
Checkout an existing branch

```
git branch -m <branch>
```

Rename a branch

Merging

- Puts forked history back together again
- Checkout the branch you want to be in and then merge the other branch in
- Git branch -d to delete target branch
- Git will try to resolve conflicts automatically



Merge Conflicts

- Git status to see which files you should manually update
- Git add git commit

- <<<<< HEAD
- =====
- >>>>> new_branch_to_merge_later

Tagging

- Reference that points to a specific point in Git history
- `git tag <tag name>`
- A tag is a branch that doesn't change
- By default, `git tag` will tag whatever commit `HEAD` is referencing

```
$ git push origin v1.4
Counting objects: 14, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (12/12), done.
Writing objects: 100% (14/14), 2.05 KiB | 0 bytes/s, done.
Total 14 (delta 3), reused 0 (delta 0)
To git@bitbucket.com:atlasbro/gittagdocs.git
 * [new tag] v1.4 -> v1.4
```

```
$ git log --pretty=oneline  
15027957951b64cf874c3557a0f3547bd83b3ff6 Merge branch  
a6b4c97498bd301d84096da251c98a07c7723e65 add update me  
0d52aaab4479697da7686c15f77a3d64d9165190 one more thin  
6d52a271eda8725415634dd79daabbc4d9b6008e Merge branch
```

Tag a non-head commit

```
git tag -a v1.2 15027957951b64cf874c3557a0f3547bd83b3f
```

```
git checkout v1.4
```

Checkout a tag

```
$ git tag  
v1  
v2  
v3  
$ git tag -d v1  
$ git tag  
v2  
v3
```

Delete a Tag

What a surprise I made a
mistake.....

- Git amend
- Git revert
- Git reset

Git Amend

Git Amend

- With `--amend` flag you can edit your most recent commit

```
$ git commit --amend
```

Add forgotten files to Commit

- Edit the file(s)
- Save the file(s)
- Stage the file(s)
- Run `git commit --amend`

Git Revert

- Git revert takes the changes in the most recent commit and does the exact opposite
 - Character is added in CommitA
 - Git revert commitA
 - New commit is created and character is deleted
 - Vice versa is true

```
new-git-project -- bash -- bash -- less -- 69x17
* db7e87a (HEAD -> master) Set page heading to "Quests & Crusades"
* 796ddb0 Merge branch 'heading-update'
|\ \
| * 4c9749e (heading-update) Set page heading to "Crusade"
* | 0c5975a Set page heading to "Quest"
|/
* 1a56a81 Merge branch 'sidebar'      Heading = Adventurous Quest
|\ \
| * f69811c (sidebar) Update sidebar with favorite movie
| * e6c65a6 Add new sidebar content
* | e014d91 (footer) Add links to social media
* | 209752a Improve site heading for SEO
* | 3772ab1 Set background color for page
|/
* 5bfe5e7 Add starting HTML structure
* 6fa5f34 Add .gitignore file
:
```

```
$ git revert <SHA-of-commit-to-revert>
```

```
new-git-project — bash — bash — bash — 69x9
richardkalehoff (master) new-git-project
$ git revert db7e87a
[master 9ec05ca] Revert "Set page heading to "Quests & Crusades"""
 1 file changed, 1 insertion(+), 1 deletion(-)
richardkalehoff (master) new-git-project
$
```

Git Reset

Relative Commit Reference

- Reference commits by - **SHA**, **tag**, **branches** and **HEAD** pointer
- Ancestry Reference
 -  ^ - indicates the parent commit
 -  ~ - indicates the first parent commit

- The parent commit
 - HEAD[^]
 - HEAD~
 - HEAD~1
- Grandparent commit
 - HEAD^{^^}
 - HEAD~2
- Great-grandparent commit
 - HEAD^{^^^}
 - HEAD~3

Difference?

-  - 1st parent
-  - 2nd Parent

```
* 9ec05ca (HEAD -> master) Revert "Set page heading to "Quests & Crusades""  
* db7e87a Set page heading to "Quests & Crusades"  
* 796ddb0 Merge branch 'heading-update'  
|\  
| * 4c9749e (heading-update) Set page heading to "Crusade"  
* | 0c5975a Set page heading to "Quest"  
|/  
* 1a56a81 Merge branch 'sidebar'  
|\  
| * f69811c (sidebar) Update sidebar with favorite movie  
| * e6c65a6 Add new sidebar content  
* | e014d91 (footer) Add links to social media  
* | 209752a Improve site heading for SEO  
* | 3772ab1 Set background color for page  
|/  
* 5bfe5e7 Add starting HTML structure  
* 6fa5f34 Add .gitignore file  
* a879849 Add header to blog  
* 94de470 Initial commit
```

1. HEAD^
2. HEAD~1
3. HEAD^^
4. HEAD~2
5. HEAD^{^}
6. HEAD~3
7. HEAD^{^}{^}2
8. HEAD~6
9. HEAD~4^2

Git Reset Command

```
$ git reset <reference-to-commit>
```

Used for:

- Move HEAD to referenced commit
- Erase commits
- Move committed changes to staging index
- Unstage committed changes

Repository



Working Directory

Staging Index

Trash

Repository

MASTER ← HEAD



git reset HEAD~1

Working Directory

Staging Index

Trash

Repository

MASTER ← HEAD
↓

d — a — 0 — 9 — f — e — 2 — a

git reset HEAD~1

Working Directory

Staging Index

Trash

3

Repository



git reset --mixed HEAD~1

Working Directory

3

Staging Index

Trash

Repository



git reset --soft HEAD~1

Working Directory

Staging Index

Trash

3

Repository

MASTER ← HEAD
↓

d — a — 0 — 9 — f — e — 2 — a

git reset --hard HEAD~1

Working Directory

Staging Index

Trash

3

Git Reset Flags

- --mixed
- --soft
- --hard

<https://www.youtube.com/watch?v=UN7ki2G2yKc>

Tips

- Git reflog
- Create a backup branch

```
* e014d91 (HEAD -> master, footer) Add links to social media
* 209752a Improve site heading for SEO
* 3772ab1 Set background color for page
* 5bfe5e7 Add starting HTML structure
* 6fa5f34 Add .gitignore file
* a879849 Add header to blog
* 94de470 Initial commit
```

- Git reset --hard HEAD~3? What about 3772ab1? Erased
- Git reset --soft HEAD^^? What about 209752a? Staging Index

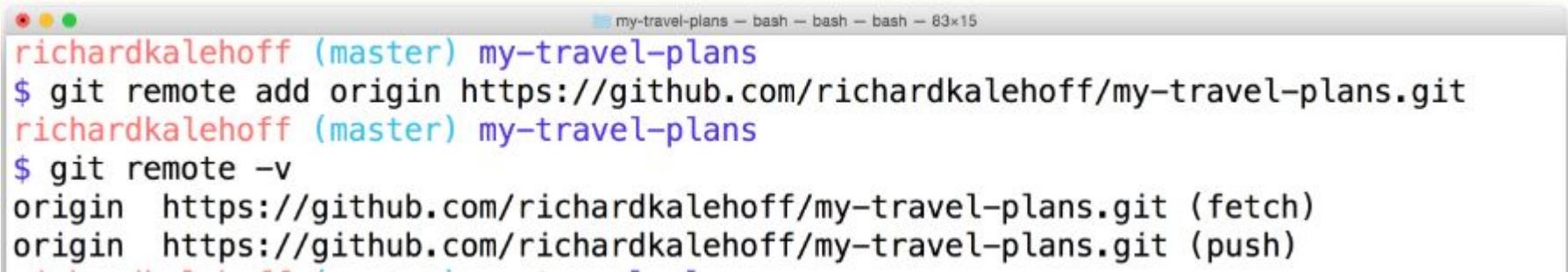
Working with Remotes

- Git remote
- Git push
- Git pull

Git Remote

Intro

- <https://www.youtube.com/watch?v=AnSIYftJnwA>
- Git remote add <shortname> <Full path of repo>
- Git remote -v



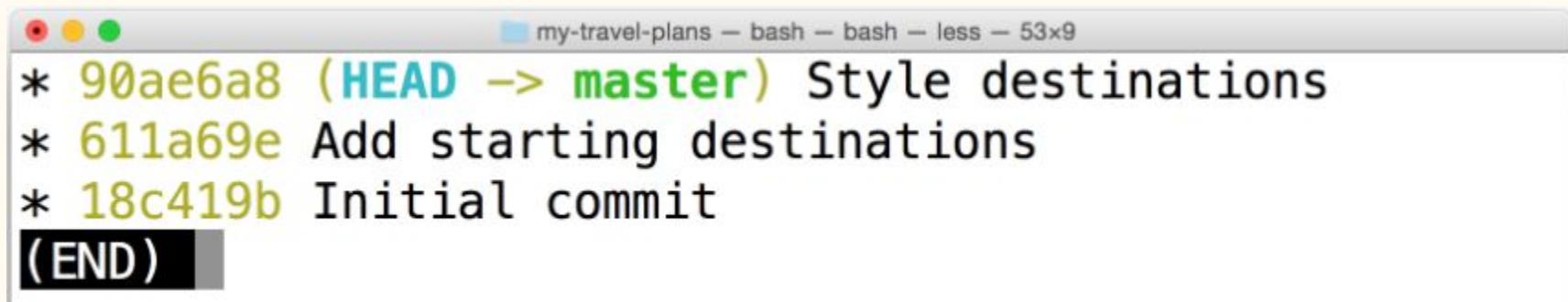
A screenshot of a terminal window titled "my-travel-plans - bash - bash - bash - 83x15". The window shows the following command-line session:

```
richardkalehoff (master) my-travel-plans
$ git remote add origin https://github.com/richardkalehoff/my-travel-plans.git
richardkalehoff (master) my-travel-plans
$ git remote -v
origin https://github.com/richardkalehoff/my-travel-plans.git (fetch)
origin https://github.com/richardkalehoff/my-travel-plans.git (push)
```

Git Push

Push changes to a remote

- <https://www.youtube.com/watch?v=21TvMEtMRys>



A screenshot of a terminal window titled "my-travel-plans - bash - bash - less - 53x9". The window displays a git log output:

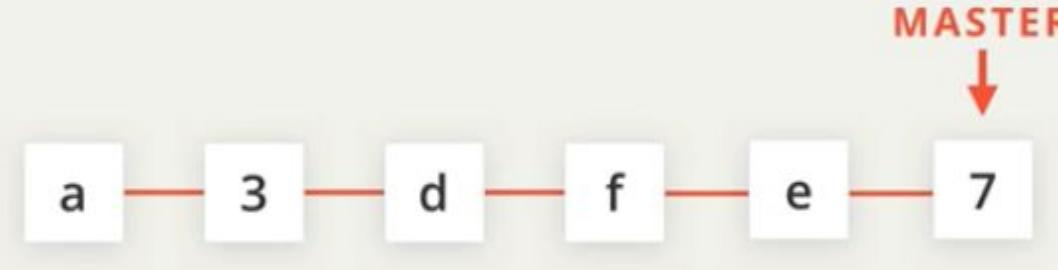
```
* 90ae6a8 (HEAD -> master) Style destinations
* 611a69e Add starting destinations
* 18c419b Initial commit
(END)
```

- Git push <short-name> <topic branch>
 - Git push origin master

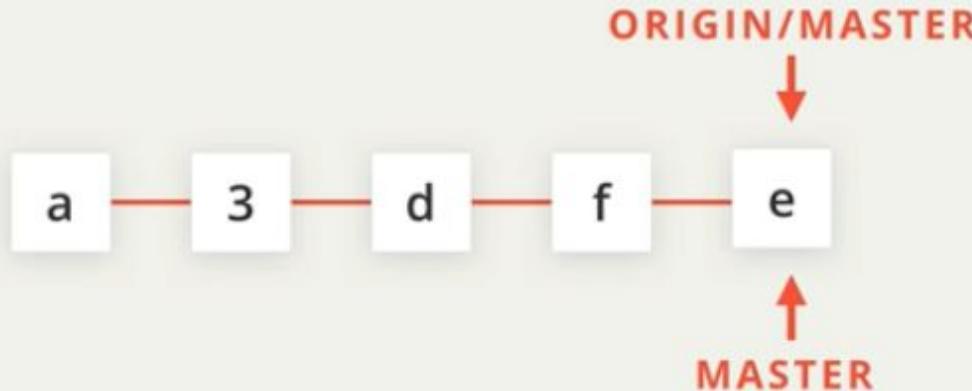
```
$ git push origin master
Username for 'https://github.com': richardkalehoff
Password for 'https://richardkalehoff@github.com':
Counting objects: 20, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (18/18), done.
Writing objects: 100% (20/20), 42.31 KiB | 0 bytes/s, done.
Total 20 (delta 4), reused 0 (delta 0)
remote: Resolving deltas: 100% (4/4), done.
To https://github.com/richardkalehoff/my-travel-plans.git
 * [new branch]      master -> master
```

Git Pull

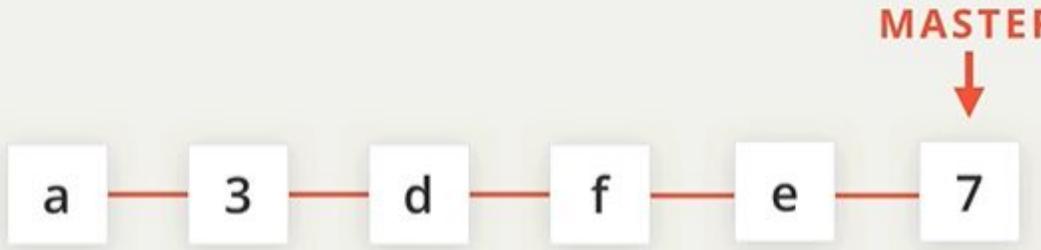
Remote
(origin)



Local



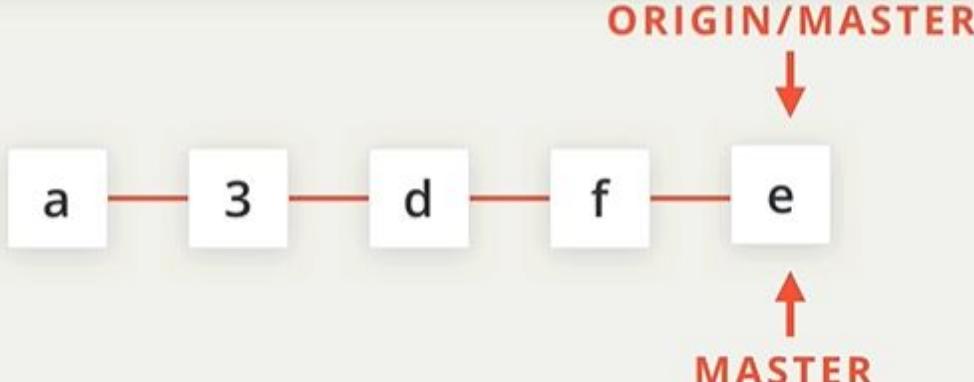
Remote
(origin)



MASTER

```
$ git pull origin master
```

Local



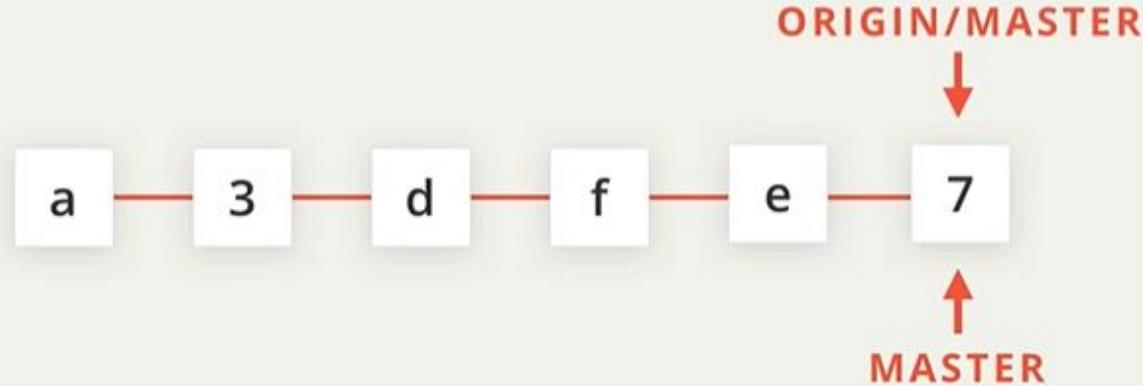
ORIGIN/MASTER

MASTER

Remote (origin)



Local

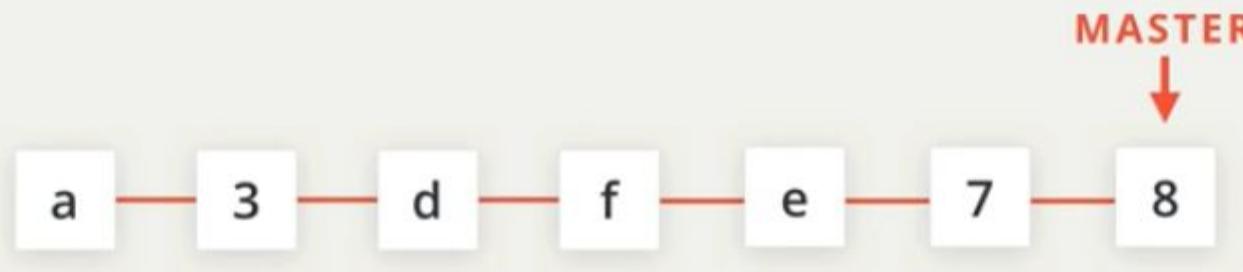


Pull changes from a remote

- <https://www.youtube.com/watch?v=MjNU2LTDVAA>
- Git pull <short-name> <topic branch>
 - Git pull origin master

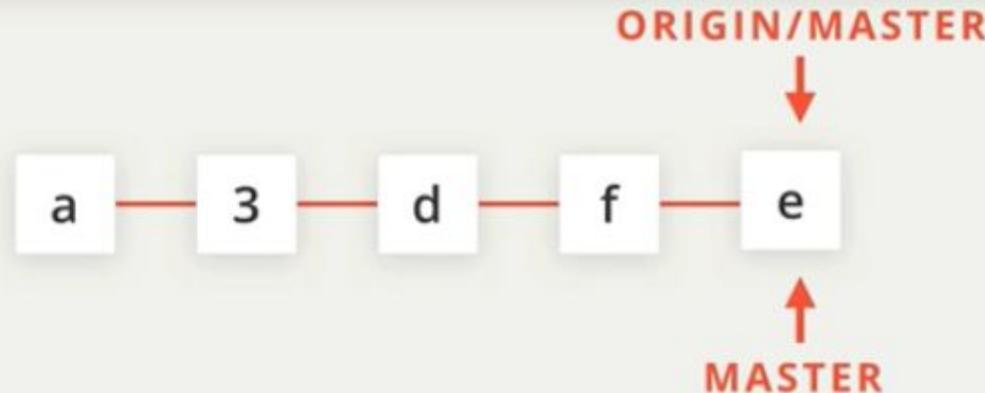
Git fetch

Remote
(origin)



```
$ git fetch origin master
```

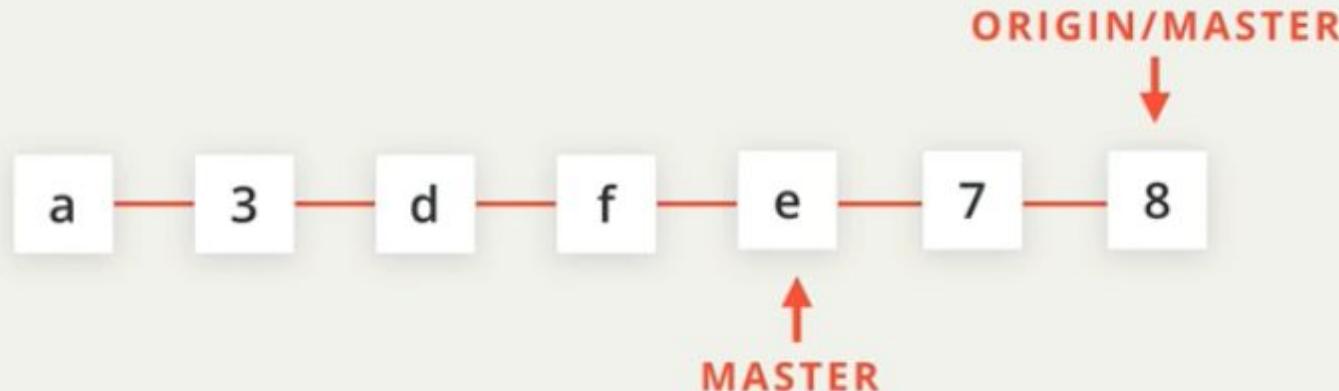
Local



Remote
(origin)



Local



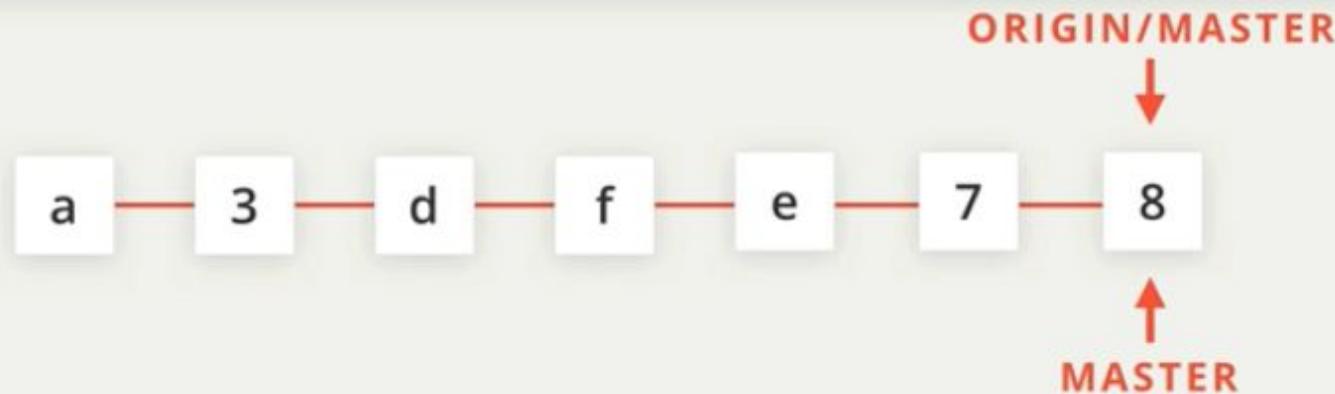
Remote
(origin)



MASTER

```
$ git merge origin/master
```

Local



ORIGIN/MASTER

MASTER

Pull v/s Fetch

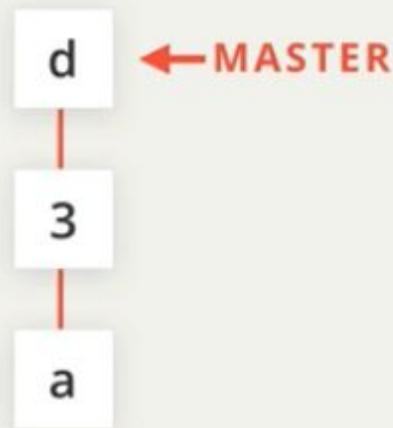
- <https://www.youtube.com/watch?v=kxXdk2HcOBo>
- The reason why -
<https://www.youtube.com/watch?v=jwyQUfE1Eqw>

Staying in Sync with a Remote Repository

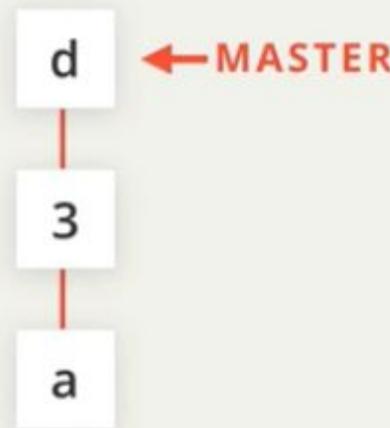
- Create a pull request
- Staying in Sync with Source Repository
- Squashing Commits

Creating a Pull Request

Forked Repository



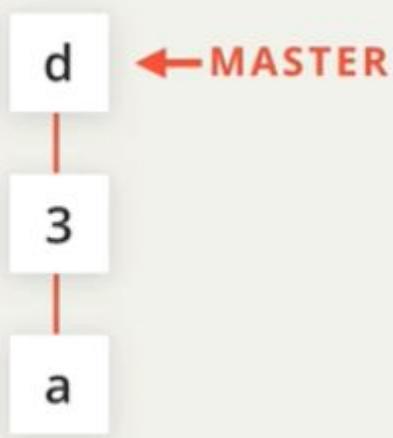
Original Repository



Local



Fork



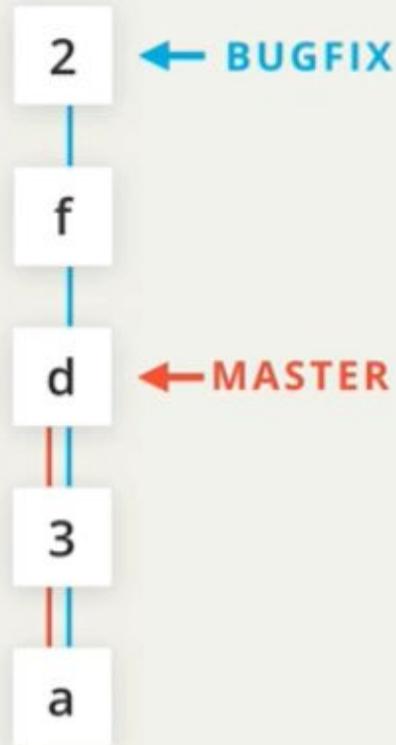
Original



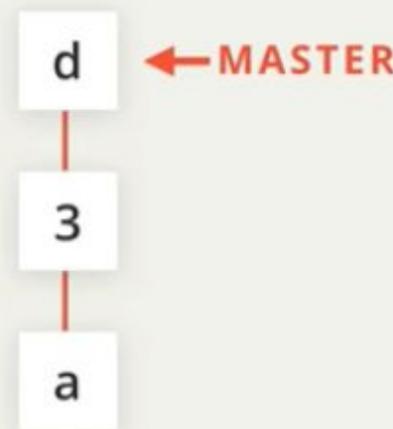
Local



Fork



Original

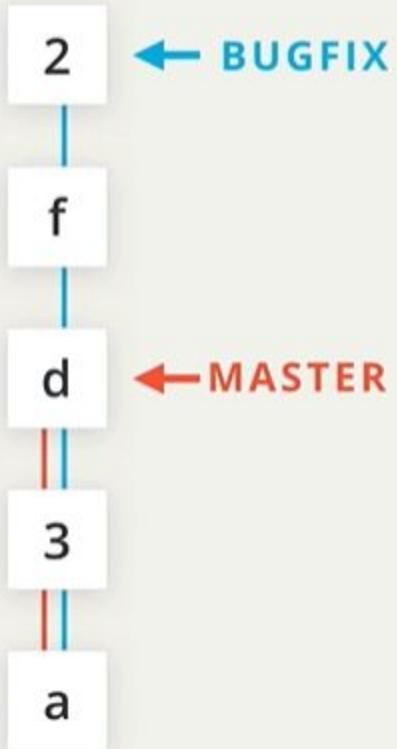


Create a pull request

- <https://www.youtube.com/watch?v=twLr9ndsf90>
- <https://www.youtube.com/watch?v=d3AGtKmHxUk>

Fork

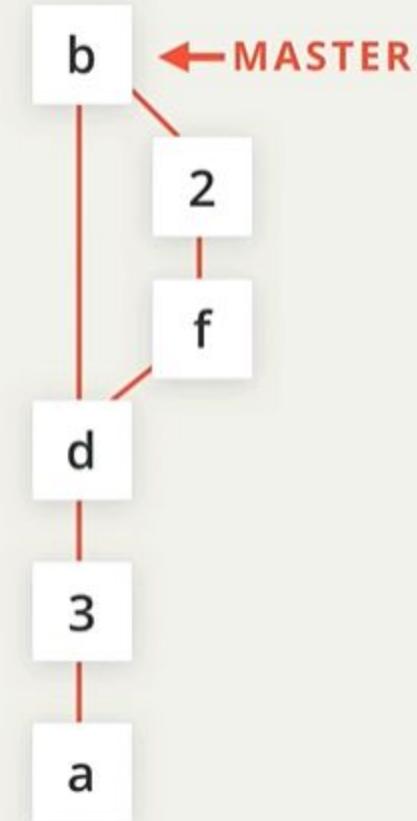
Local



Fork

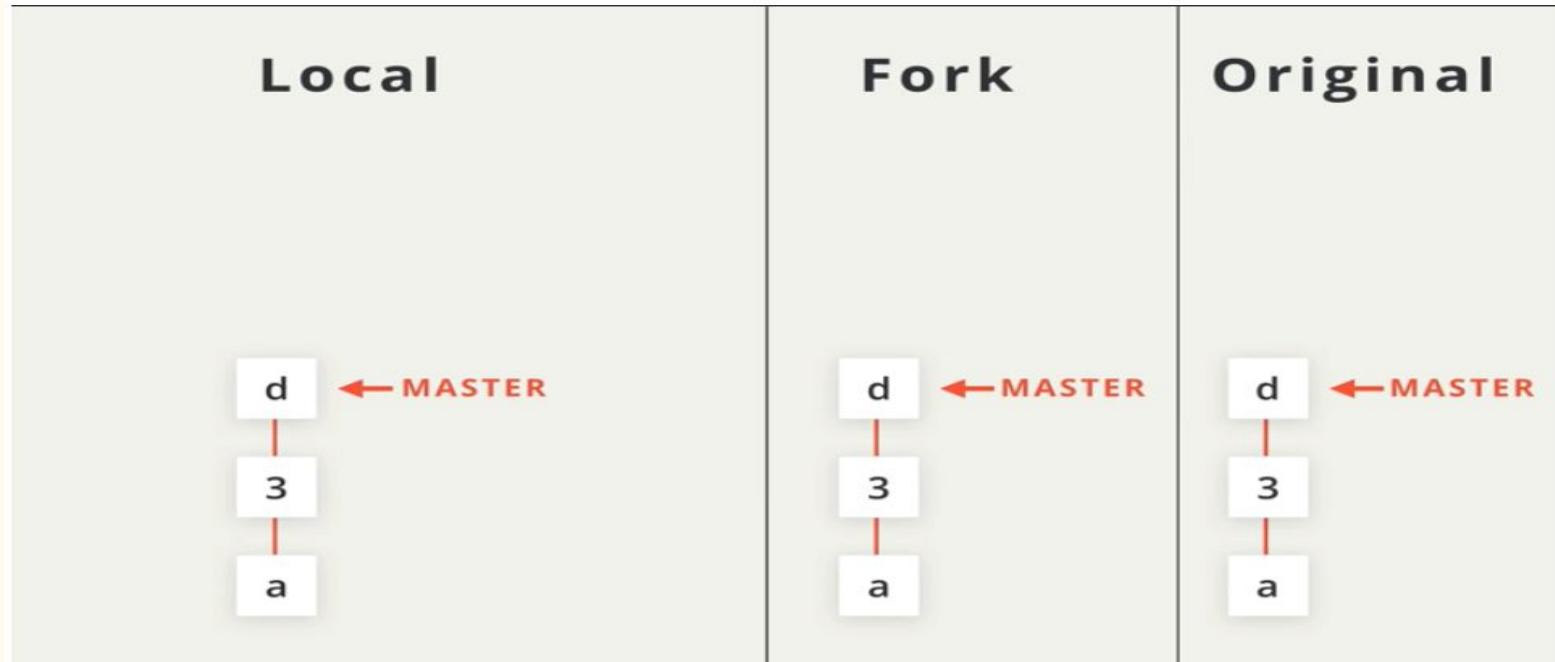


Original

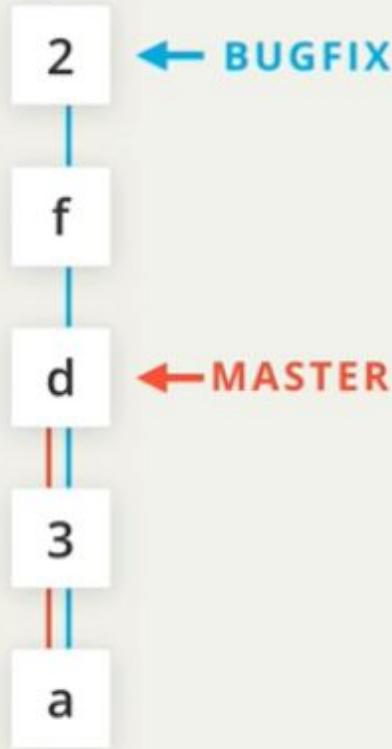


Staying in sync w/
Remote Repository

Staying in sync with remote repo



Local



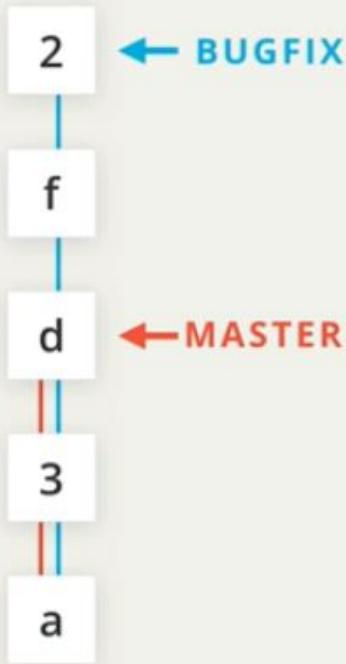
Fork



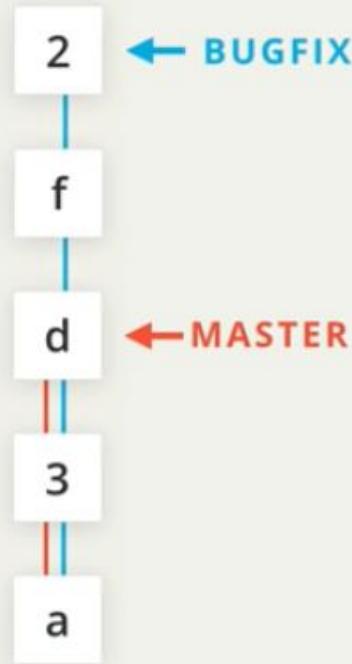
Original



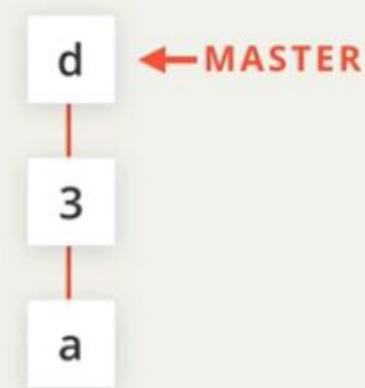
Local



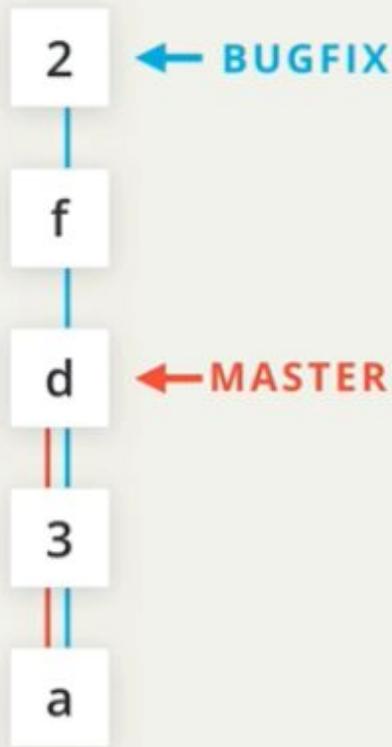
Fork



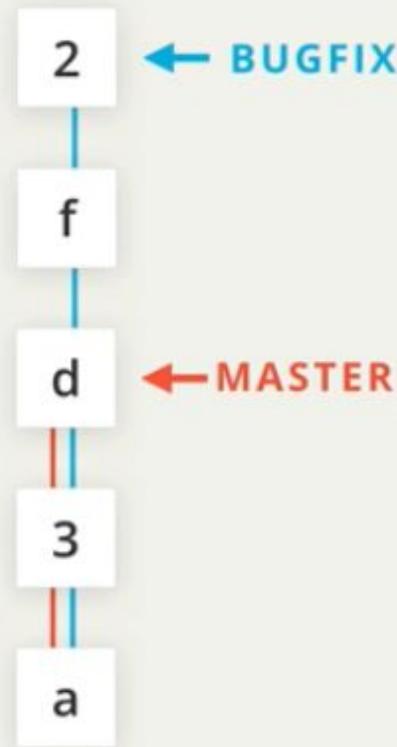
Original



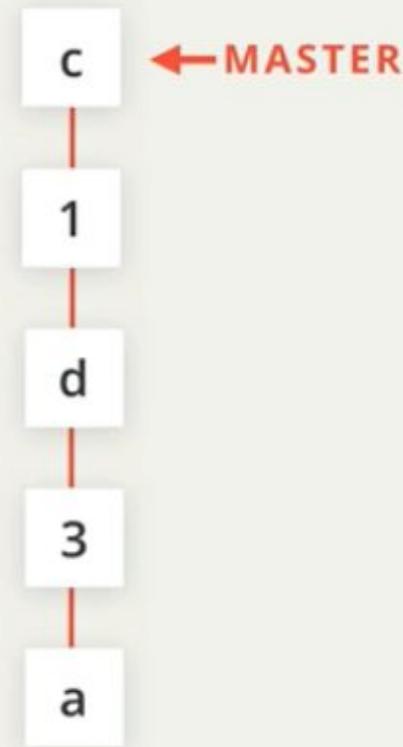
Local



Fork



Original



Fork

ORIGIN

Original

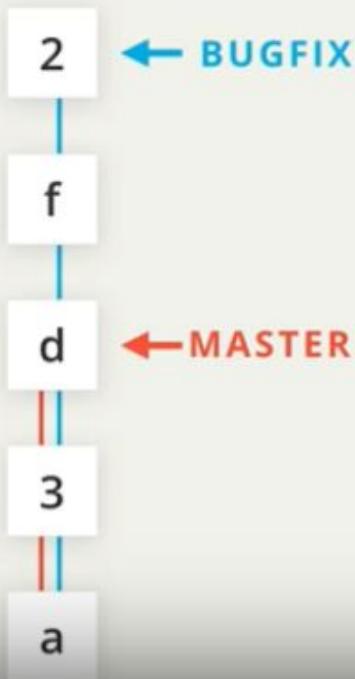
UPSTREAM

Local

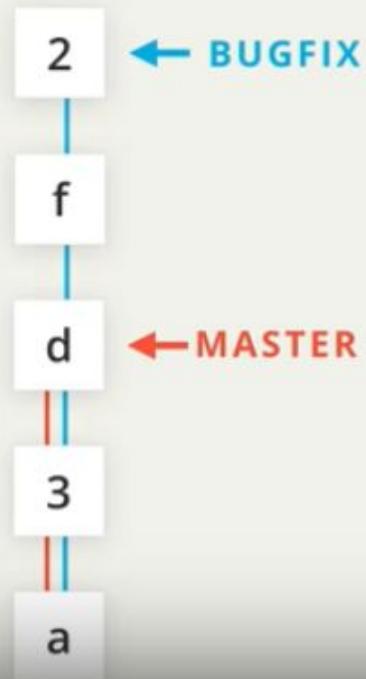


- Git remote add upstream <URL of src repo>
- Git remote rename <oldname> <newname>

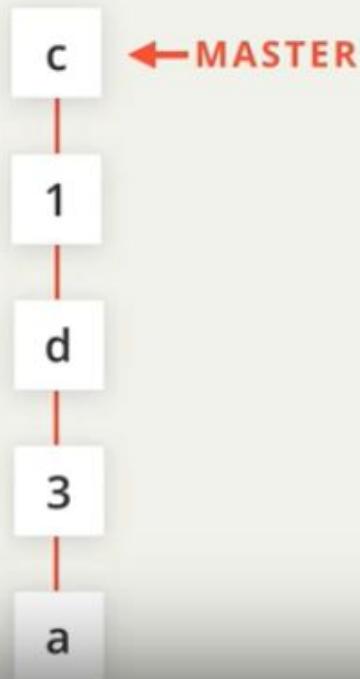
Local



Fork (origin)

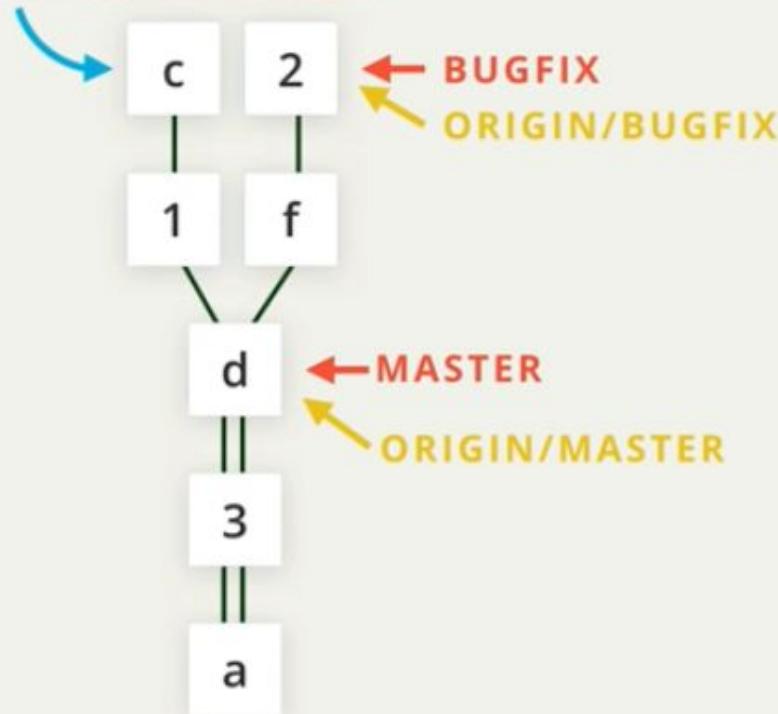


Original (upstream)



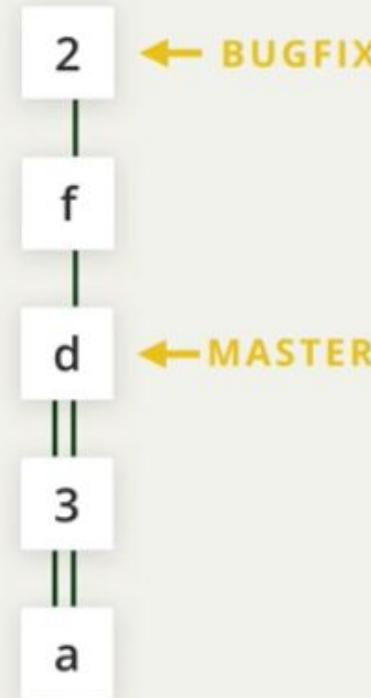
Local

UPSTREAM/MASTER



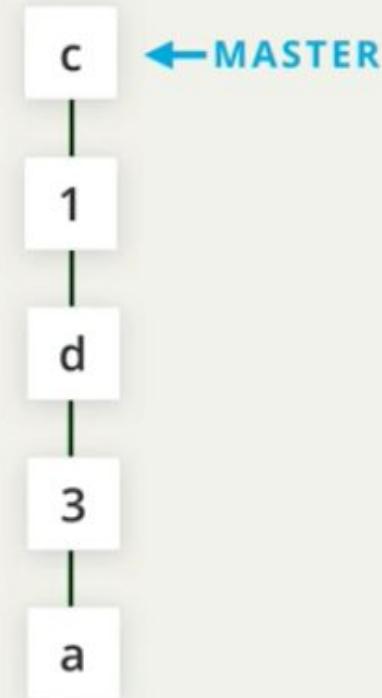
Fork

(origin)

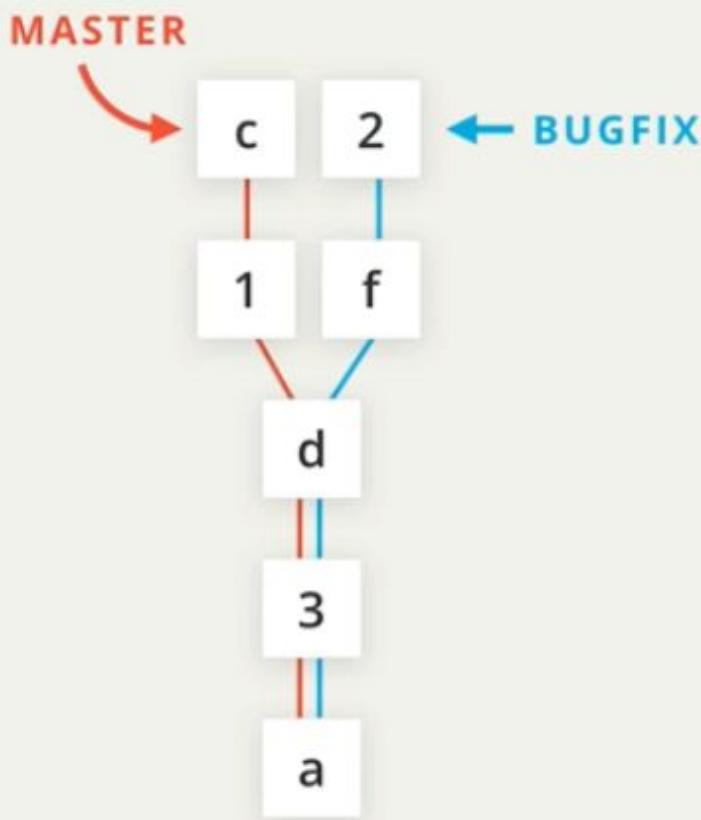


Original

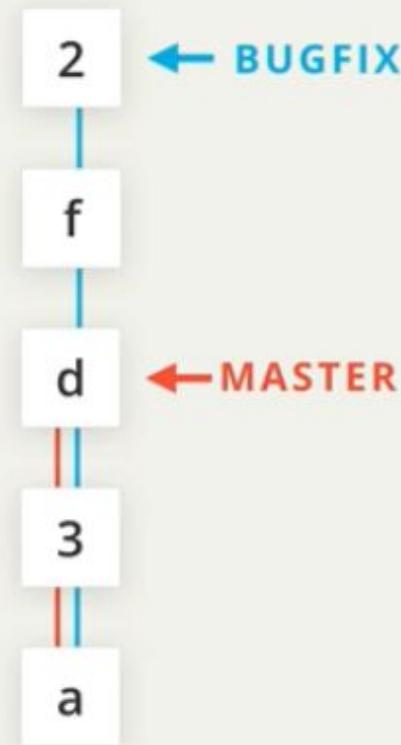
(upstream)



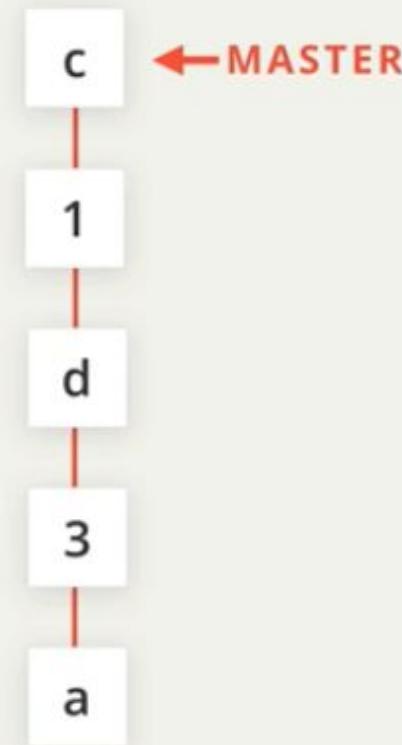
Local



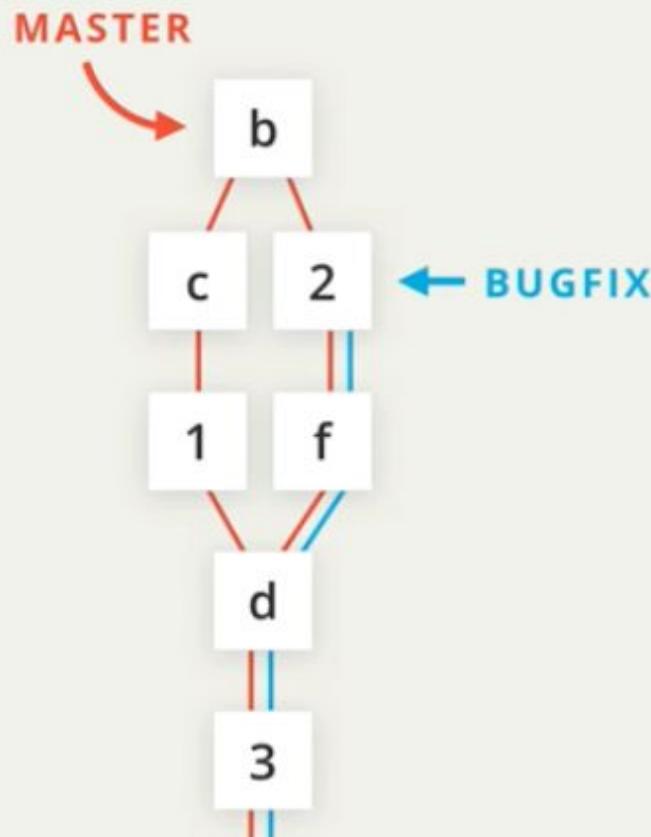
Fork (origin)



Original (upstream)



Local



Fork

(origin)



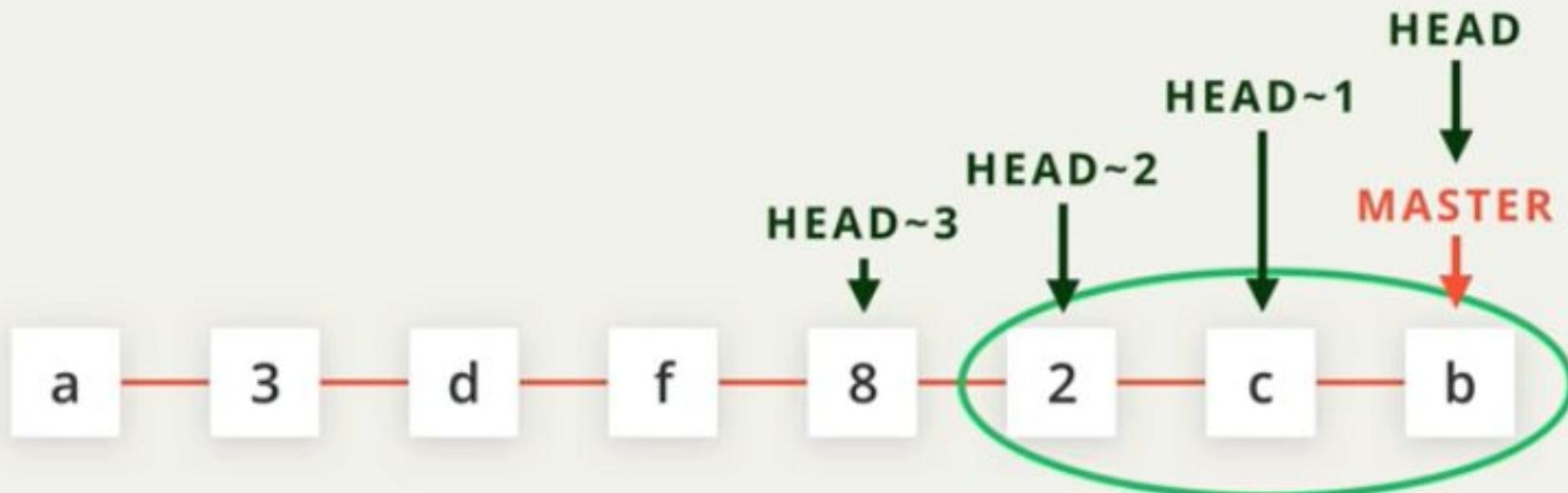
Original

(upstream)



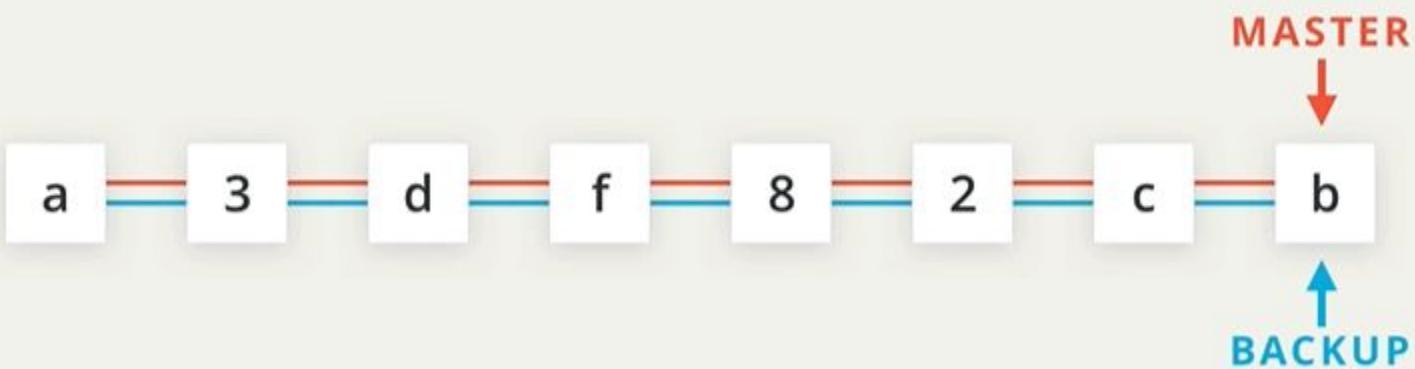
Squashing Commits

```
$ git rebase -i HEAD~3
```

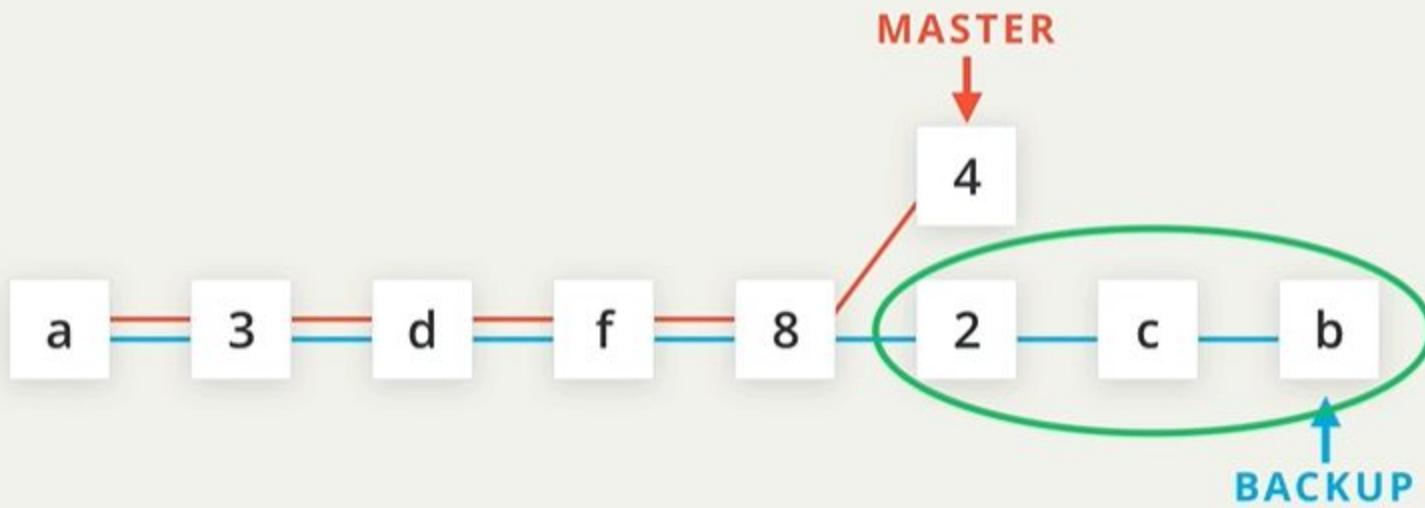


```
$ git rebase -i HEAD~3
```





```
$ git rebase -i HEAD~3
```



Atom File Edit View Selection Find Packages Window Help

git-rebase-todo — /Public/GitHub Course - Richard/Projects/course-collaboration-travel-plans/git/rebase-merge

```
rebase-merge git-rebase-todo
```

```
1 pick 8aa9efb Add destination to Florida
2 pick d968543 Add destination to Paris
3 pick ac5ce47 Add destination to Scotland
4
5 # Rebase b2d0353..ac5ce47 onto b2d0353 (3 commands)
6 #
7 # Commands:
8 # p, pick = use commit
9 # r, reword = use commit, but edit the commit message
10 # e, edit = use commit, but stop for amending
11 # s, squash = use commit, but meld into previous commit
12 # f, fixup = like "squash", but discard this commit's log message
13 # x, exec = run command (the rest of the line) using shell
14 # d, drop = remove commit
15 #
```

Atom File Edit View Selection Find Packages Window Help

git-rebase-todo — ~/Public/GitHub/Course - Richard/Projects/course-collaboration-travel-plans/git/rebase-merge

rebase-merge git-rebase-todo

```
pick 8aa9efb Add destination to Florida
s d968543 Add destination to Paris
s ac5ce47 Add destination to Scotland
#
# Rebase b2d0353..ac5ce47 onto b2d0353 (3 commands)
#
# Commands:
# p, pick = use commit
# r, reword = use commit, but edit the commit message
# e, edit = use commit, but stop for amending
# s, squash = use commit, but meld into previous commit
# f, fixup = like "squash", but discard this commit's log message
# x, exec = run command (the rest of the line) using shell
# d, drop = remove commit
#
```

```
Atom  File  Edit  View  Selection  Find  Packages  Window  Help
git-rebase-todo — ~/Public/GitHub Course - Richard/Projects/course-collaboration-travel-plans/git/rebase-merge

rebase-merge git-rebase-todo

1  r 8aa9efb Add destination to Florida
2  s d968543 Add destination to Paris
3  s ac5ce47 Add destination to Scotland
4
5  # Rebase b2d0353..ac5ce47 onto b2d0353 (3 commands)
6  #
7  # Commands:
8  # p, pick = use commit
9  # r, reword = use commit, but edit the commit message
10 # e, edit = use commit, but stop for amending
11 # s, squash = use commit, but meld into previous commit
12 # f, fixup = like "squash", but discard this commit's log message
13 # x, exec = run command (the rest of the line) using shell
14 # d, drop = remove commit
15 #
```

git

COMMIT_EDITMSG

```
1 Add Richard's destinations
2
3 # Please enter the commit message for your changes. Lines starting
4 # with '#' will be ignored, and an empty message aborts the commit.
5 #
6 # Date:      Tue Apr 11 17:53:06 2017 -0700
7 #
8 # interactive rebase in progress; onto b2d0353
9 # Last command done (1 command done):
10 #    r 8aa9efb Add destination to Florida
11 # Next commands to do (2 remaining commands):
12 #    s d968543 Add destination to Paris
13 #    s ac5ce47 Add destination to Scotland
14 # You are currently editing a commit while rebasing branch 'include-richards-destination'
15 #
16 # Changes to be committed:
17 #   modified:  css/app.css
18 #   modified:  index.html
19 #
20
```

Atom File Edit View Selection Find Packages Window Help

COMMIT_EDITMSG — ~/Public/GitHub/Course - Richard/Projects/course-collaboration-travel-plans/.git

.git COMMIT_EDITMSG

```
1 # This is a combination of 3 commits.
2 # This is the 1st commit message:
3 Add Richard's destinations
4
5 # This is the commit message #2:
6
7 Add destination to Paris
8
9 # This is the commit message #3:
10
11 Add destination to Scotland
12
13 # Please enter the commit message for your changes. Lines starting
14 # with '#' will be ignored, and an empty message aborts the commit.
15 #
16 # Date:      Tue Apr 11 17:53:06 2017 -0700
17 #
18 # interactive rebase in progress; onto b2d0353
```

```
1 Add Richard's destinations
2
3 # Please enter the commit message for your changes. Lines starting
4 # with '#' will be ignored, and an empty message aborts the commit.
5 #
6 # Date:      Tue Apr 11 17:53:06 2017 -0700
7 #
8 # interactive rebase in progress; onto b2d0353
9 # Last commands done (3 commands done):
10 #   s d968543 Add destination to Paris
11 #   s ac5ce47 Add destination to Scotland
12 # No commands remaining.
13 # You are currently editing a commit while rebasing branch 'include-richards-destination'
14 #
15 # Changes to be committed:
16 #   modified:  css/app.css
17 #   modified:  index.html
18 #
19
```

```
$ git log --oneline --graph --decorate --all
* f69a327 (HEAD -> include-richards-destinations) Add Richard's destinations
| * ac5ce47 (origin/include-richards-destinations) Add destination to Scotland
| * d968543 Add destination to Paris
| * 8aa9efb Add destination to Florida
|/
* b2d0353 (origin/master, origin/HEAD, master) Add animation to destination headings
* 1204be0 Style destinations
* 7562e21 Add starting destinations
* 5e9b201 Initial commit
```

```
$ git push origin include-richards-destinations
To https://github.com/richardkalehoff/course-collaboration-travel-plans.git
 ! [rejected]      include-richards-destinations -> include-richards-destinations (non-fast-forward)
error: failed to push some refs to 'https://github.com/richardkalehoff/course-collaboration-travel-plans.
git'
hint: Updates were rejected because the tip of your current branch is behind
hint: its remote counterpart. Integrate the remote changes (e.g.
hint: 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

- Git push -f origin include-richards-destination

Rebase Command

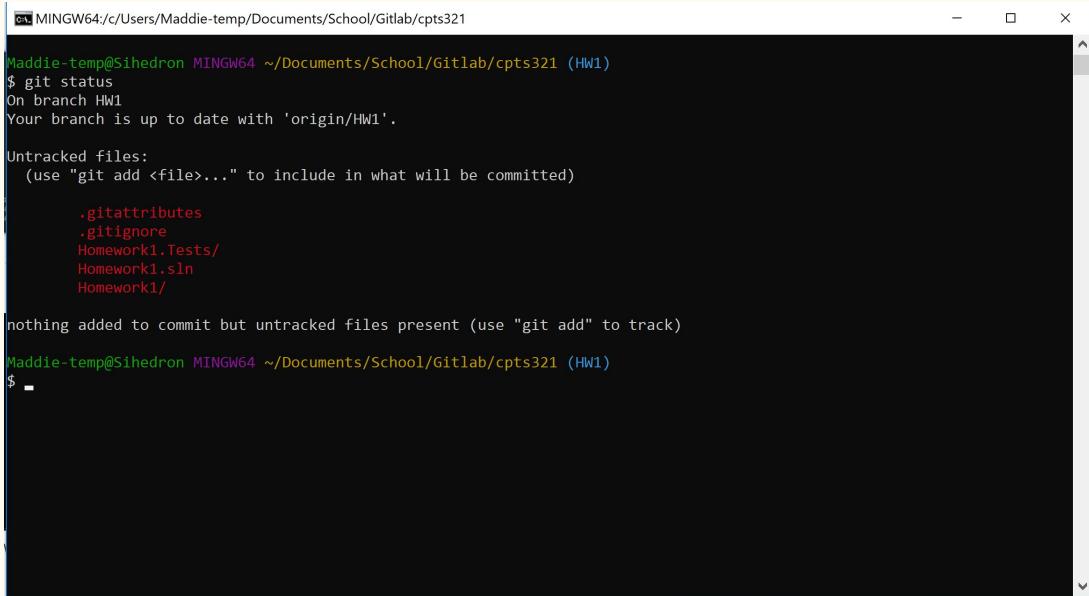
- Use **p** or **pick** - to keep the commit as is
- Use **r** or **reword** - to keep the commit's content but alter the commit message
- Use **e** or **edit** - to keep the commit's content but stop before committing so that you can:
 - Add new content or files
 - Remove content or files
 - Alter the content that was going to be committed
- Use **s** or **squash** - to combine commit's changes into the previous commit (the commit above)
- Use **f** or **fixup** - to combine commit's change into the previous one but drop the commit message
- Use **x** or **exec** - to run a shell command
- Use **d** or **drop** - to delete the commit

Tools - Bash, Plugins, and a GUI

Git Bash

What is it?

- A command-line interface to use Git
- Styled after Linux command lines
- “The standard” way to use Git



```
MINGW64:/c/Users/Maddie-temp/Documents/School/Gitlab/cpts321
Maddie-temp@Sihedron MINGW64 ~/Documents/School/Gitlab/cpts321 (HW1)
$ git status
On branch HW1
Your branch is up to date with 'origin/HW1'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    .gitattributes
    .gitignore
    Homework1.Tests/
    Homework1.sln
    Homework1/

nothing added to commit but untracked files present (use "git add" to track)

Maddie-temp@Sihedron MINGW64 ~/Documents/School/Gitlab/cpts321 (HW1)
$ -
```

Git Bash - Pros and Cons

Pros:

- Extremely powerful tool
- Fast and flexible
- Similar to Linux shell
- Looks really cool

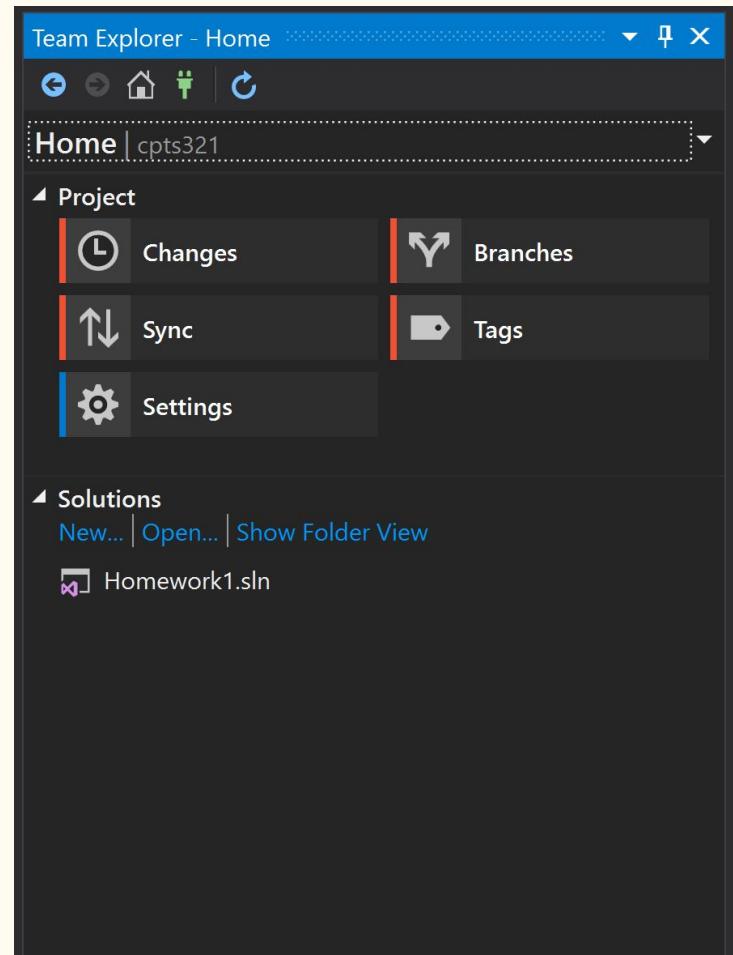
Cons:

- Very steep learning curve
- Not easy to visually parse
- Not user friendly

Visual Studio Team Explorer

What is it?

- An integrated tool to use Git
- Works with any Git service (Github, Gitlab, etc.)
- Everything in one area
- This student's go-to!



Team Explorer - Pros and Cons

Pros:

- Integrated with VS
- Simple, intuitive interface
- Can view everything graphically

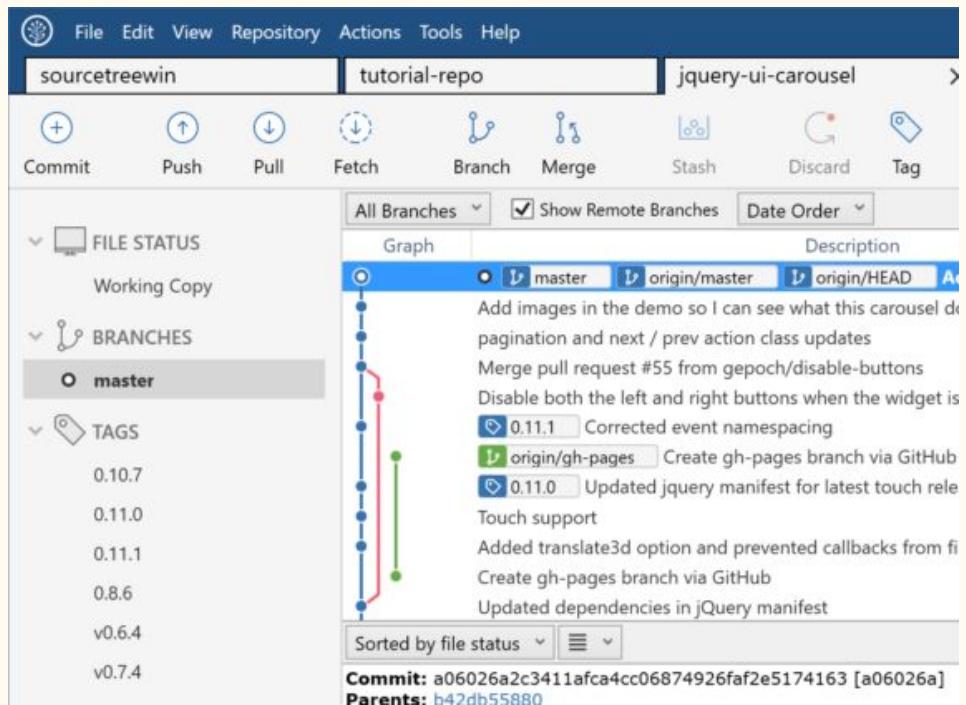
Cons:

- Only in VS, and not in VS Community
- Requires a bit of setup
- Takes a bit longer to use than Bash
(clicking is slower than typing)

Other GUI applications

What is it?

- Lots of other third-party applications
- Provide an easy-to-use interface
- Popular ones include Sourcetree (pictured), Github desktop, SmartGit, and more



GUI Applications - Pros and Cons

Pros:

- Tons of choices
- GUIs are easier and more user friendly
- Language/IDE agnostic - use them regardless of the language

Cons:

- Have to install/set up
- Some cost money
- Yet another tool to learn

Expired Passwords

Your emotions:

- You're angry
- You're upset
- You're sad
- What can you do?

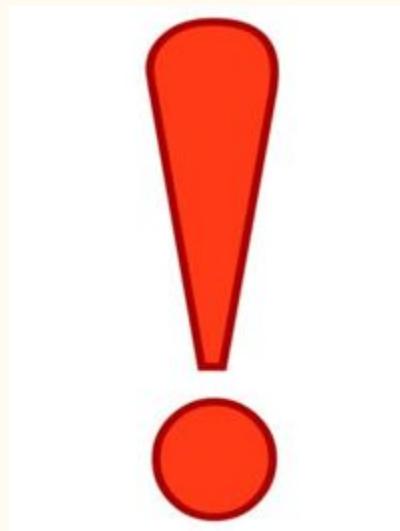


You can call the help desk

And listen to a boring person on the
other line help you fix it.



OR



You can reset it on your own.

To start off:

- Open Bash
- Type ssh
- First initial then last name which is also your gitlab username.
- ssh 1-10
- eecs.wsu.edu



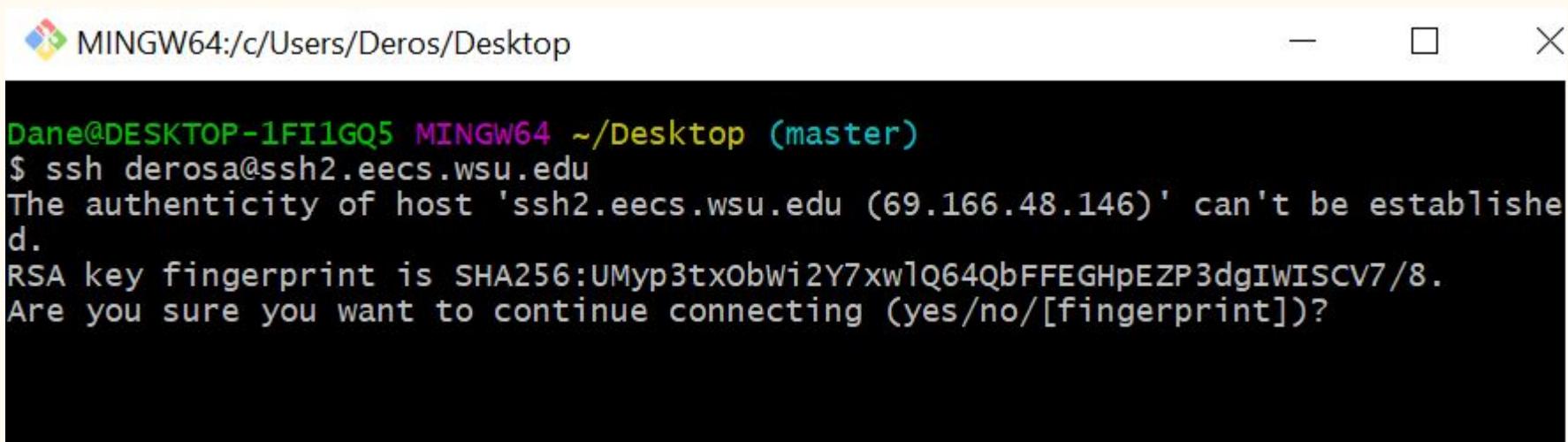
MINGW64:/c/Users/Deros/Desktop

```
Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop (master)
$ ssh derosa@ssh2.eecs.wsu.edu|
```

RSA Fingerprint

Whoa!

- Type: Yes

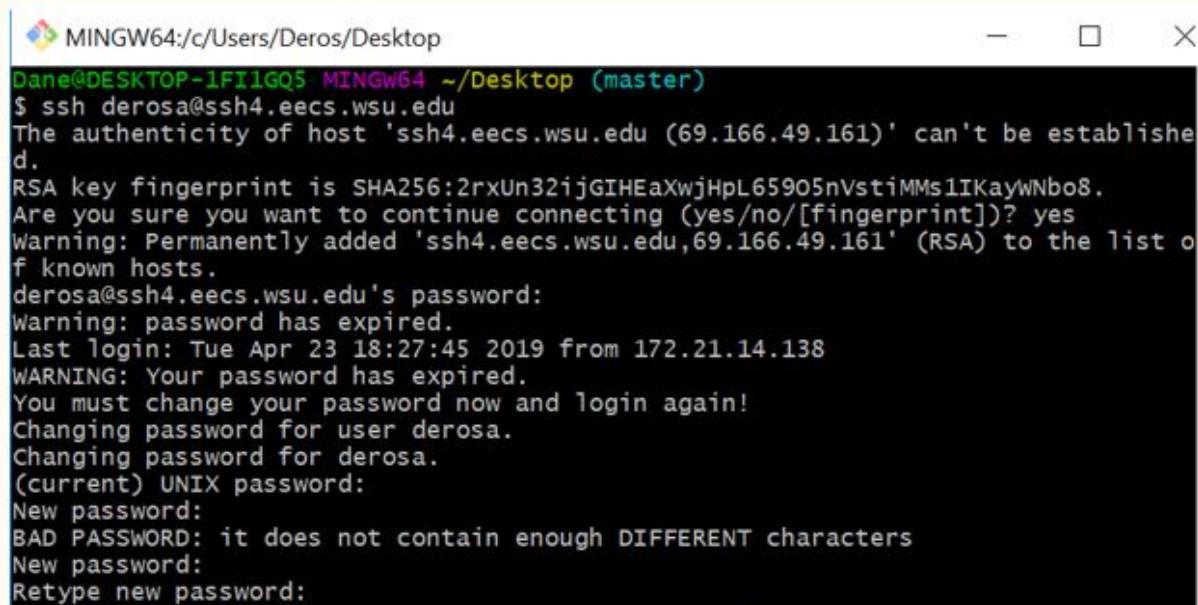


The screenshot shows a terminal window titled "MINGW64:/c/Users/Deros/Desktop". The terminal output is as follows:

```
Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop (master)
$ ssh derosa@ssh2.eecs.wsu.edu
The authenticity of host 'ssh2.eecs.wsu.edu (69.166.48.146)' can't be established.
RSA key fingerprint is SHA256:UMyp3txObWi2Y7xw1Q64QbFFEGHpEZP3dgIWISCV7/8.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

Changing your password

- Your current UNIX password is your previously known expired password.
- Enter your new password twice.

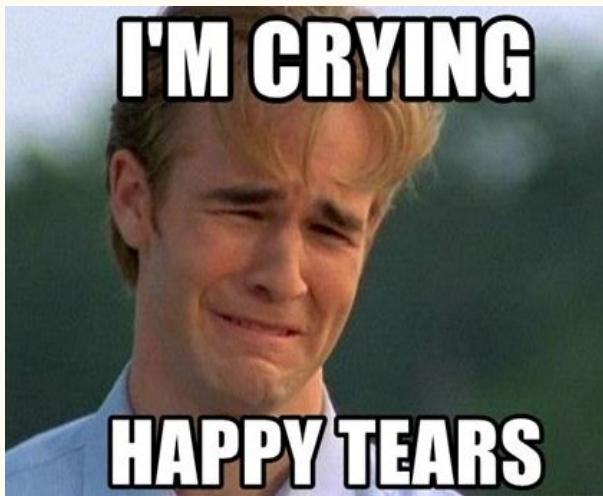


A screenshot of a terminal window titled "MINGW64:/c/Users/Deros/Desktop". The window shows a command-line session where a user named "derosa" is attempting to log in via SSH to a host at "ssh4.eecs.wsu.edu". The session starts with the user being prompted for their password, which is noted as having expired. The user is then prompted to change their password, and they enter a new password that fails the password strength check because it does not contain enough different characters. The terminal window has standard window controls (minimize, maximize, close) in the top right corner.

```
Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop (master)
$ ssh derosa@ssh4.eecs.wsu.edu
The authenticity of host 'ssh4.eecs.wsu.edu (69.166.49.161)' can't be established.
RSA key fingerprint is SHA256:2rxUn32ijGIHEaXwjHpL659O5nVstiMMs1IKayWNbo8.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ssh4.eecs.wsu.edu,69.166.49.161' (RSA) to the list of known hosts.
derosa@ssh4.eecs.wsu.edu's password:
Warning: password has expired.
Last login: Tue Apr 23 18:27:45 2019 from 172.21.14.138
WARNING: Your password has expired.
You must change your password now and login again!
Changing password for user derosa.
Changing password for derosa.
(current) UNIX password:
New password:
BAD PASSWORD: it does not contain enough DIFFERENT characters
New password:
Retype new password:
```

SUCCESS!

- We have changed our password and now things are fine but we are still behind on homework.



```
MINGW64:/c/Users/Deros/Desktop
Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop (master)
$ ssh derosa@ssh4.eecs.wsu.edu
The authenticity of host 'ssh4.eecs.wsu.edu (69.166.49.161)' can't be established.
RSA key fingerprint is SHA256:2rxUn32ijGIHeaXwjHpL65905nVstiMMs1IKayWNbo8.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ssh4.eecs.wsu.edu,69.166.49.161' (RSA) to the list o
f known hosts.
derosa@ssh4.eecs.wsu.edu's password:
Warning: password has expired.
Last login: Tue Apr 23 18:27:45 2019 from 172.21.14.138
WARNING: Your password has expired.
You must change your password now and login again!
Changing password for user derosa.
Changing password for derosa.
(current) UNIX password:
New password:
BAD PASSWORD: it does not contain enough DIFFERENT characters
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
Connection to ssh4.eecs.wsu.edu closed.
```

Let's put it all together

Step 1: Setup Repo

Create a new project

Blank project Create from template Import project

Project name
 x

Project URL

Project slug

Want to house several dependent projects under the same namespace? [Create a group](#).

Project description (optional)

Description format

Visibility Level ?

Private
Project access must be granted explicitly to each user.

Internal
The project can be accessed by any logged in user.

Public
The project can be accessed without any authentication.

Initialize repository with a README
Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.

Cancel Create project

CPTS_321 Tester Project ID: 2079 Clone

[Add license](#)

The repository for this project is empty

You can create files directly in GitLab using one of the following options.

New file Add README Add CHANGELOG Add CONTRIBUTING

Command line instructions

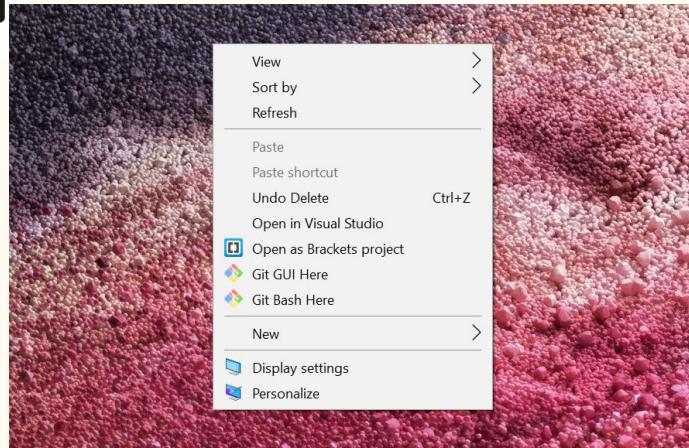
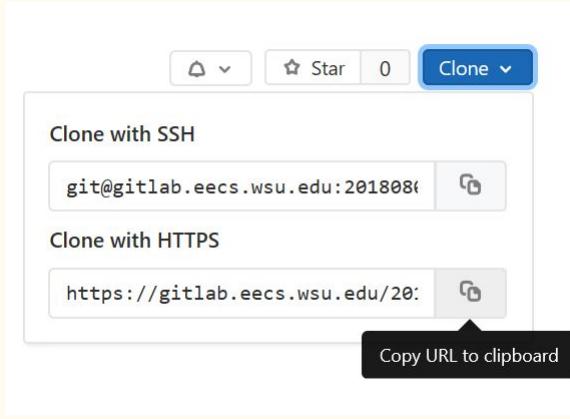
You can also upload existing files from your computer using the instructions below.

Git global setup

```
git config --global user.name "derosa"  
git config --global user.email "derosa@eecs.wsu.edu"
```

Step 2: Clone Repo

Click Clone: and we are using HTTPS



```
MINGW64:/c/Users/Deros/Desktop
git
Dane@DESKTOP-1F11GQ5 MINGW64 ~/Desktop (master)
$ git clone https://gitlab.eecs.wsu.edu/20180816/cpts_321-tester.git
Cloning into 'cpts_321-tester'...
warning: You appear to have cloned an empty repository.

Dane@DESKTOP-1F11GQ5 MINGW64 ~/Desktop (master)
$
```

Step 3: Actions

Move Hello World into folder.

```
Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop/cpts_321-tester (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    Hello_World_One/
    Test/
nothing added to commit but untracked files present (use "git add" to track)

Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop/cpts_321-tester (master)
$ git add Hello_World_One

Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop/cpts_321-tester (master)
$ git commit -m 'Uploading For First Time'
```

```
Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop/cpts_321-tester (master)
$ git push
Enumerating objects: 30, done.
Counting objects: 100% (30/30), done.
Delta compression using up to 8 threads
Compressing objects: 100% (22/22), done.
Writing objects: 100% (29/29), 182.05 KiB | 1.65 MiB/s, done.
Total 29 (delta 1), reused 0 (delta 0)
To https://gitlab.eecs.wsu.edu/2018080816/cpts_321-tester.git
  db242f4..52c33ce  master -> master
```

Step 4: Editing

We edited Hello World One and Pushed it back

```
Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop/cpts_321-tester (master)
$ git add Hello_World_One

Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop/cpts_321-tester (master)
$ git commit -m 'Added another Hello World'
[master 2434a65] Added another Hello World
 9 files changed, 10 insertions(+), 10 deletions(-)
 rewrite Hello_World_One/.vs/Hello_World_One/v16/.suo (60%)
 rewrite Hello_World_One/Hello_World_One/obj/Debug/netcoreapp2.1>Hello_World_One.csprojAssemblyReference.cache (99%)

Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop/cpts_321-tester (master)
$ git push
Enumerating objects: 42, done.
Counting objects: 100% (42/42), done.
Delta compression using up to 8 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (22/22), 4.34 KiB | 126.00 KiB/s, done.
Total 22 (delta 9), reused 0 (delta 0)
To https://gitlab.eecs.wsu.edu/2018080816/cpts_321-tester.git
 52c33ce..2434a65  master -> master
```

Step 5: Create & Push to Branch

Let's open bash in our folder again

```
Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop/cpts_321-tester (master)
$ git pull
Already up to date.
```

```
Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop/cpts_321-tester (master)
$ git checkout -b 'Demo_Tester'
Switched to a new branch 'Demo_Tester'
```

```
Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop/cpts_321-tester (Demo_Tester)
$ git status
On branch Demo_Tester
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    Test/
nothing added to commit but untracked files present (use "git add" to track)
```

```
Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop/cpts_321-tester (Demo_Tester)
$ git add Test
```

```
Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop/cpts_321-tester (Demo_Tester)
$ git commit -m 'Uploading Test to Demo_Tester'
[Demo_Tester b6fdc1b] Uploading Test to Demo_Tester
 21 files changed, 1248 insertions(+)
```

```
Dane@DESKTOP-1FI1GQ5 MINGW64 ~/Desktop/cpts_321-tester (Demo_Tester)
$ git push --set-upstream origin Demo_Tester
Enumerating objects: 37, done.
Counting objects: 100% (37/37), done.
Delta compression using up to 8 threads
Compressing objects: 100% (28/28), done.
Writing objects: 100% (36/36), 173.13 KiB | 1.37 MiB/s, done.
Total 36 (delta 2), reused 0 (delta 0)
remote:
remote: To create a merge request for Demo_Tester, visit:
remote:   https://gitlab.eecs.wsu.edu/2018080816/cpts_321-tester/merge_requests/
new?merge_request%5Bsource_branch%5D=Demo_Tester
remote:
To https://gitlab.eecs.wsu.edu/2018080816/cpts_321-tester.git
 2434a65..b6fdc1b  Demo_Tester -> Demo_Tester
Branch 'Demo_Tester' set up to track remote branch 'Demo_Tester' from 'origin'.
```