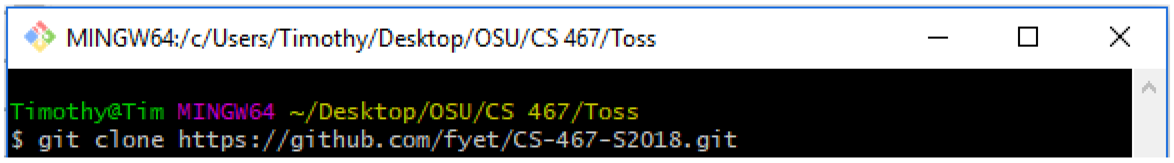
Git Bash & GitHub Guide

Clone Our Repository

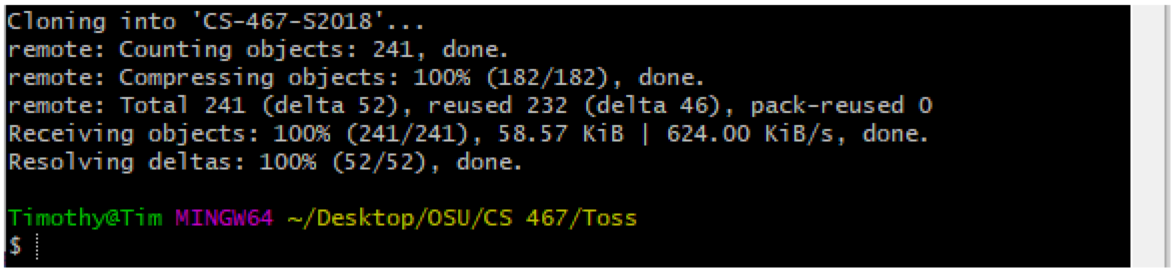
Abstract: This command is use once to place a copy of our repository onto our local machines

Command: `git clone https://github.com/fyet/CS-467-S2018.git`



```
Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/Toss
$ git clone https://github.com/fyet/CS-467-S2018.git
```

After hitting enter:

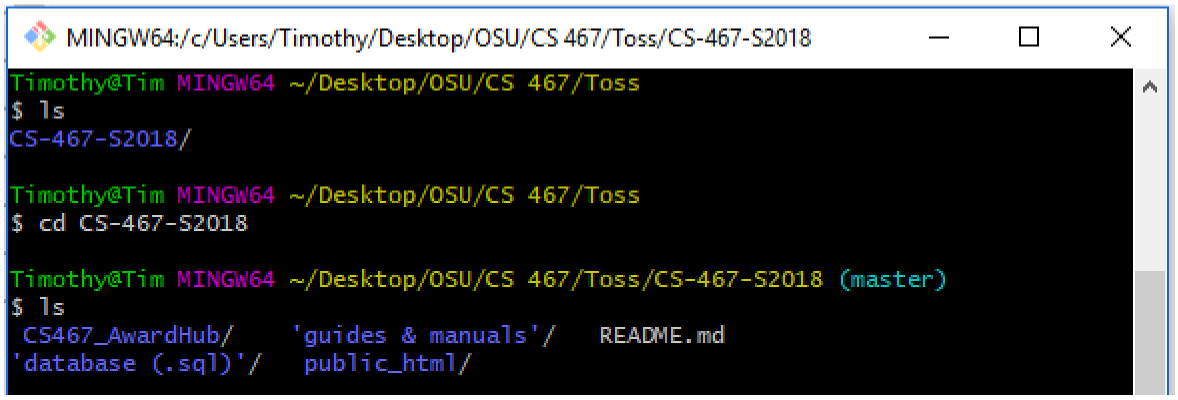


```
Cloning into 'CS-467-S2018'...
remote: Counting objects: 241, done.
remote: Compressing objects: 100% (182/182), done.
remote: Total 241 (delta 52), reused 232 (delta 46), pack-reused 0
Receiving objects: 100% (241/241), 58.57 KiB | 624.00 KiB/s, done.
Resolving deltas: 100% (52/52), done.

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/Toss
$
```

Navigate to Folder

Abstract: This is pretty trivial, but I will add it anyway just for the sake of being thorough. After the repository is cloned, we will need to step inside of the directory to work in it

Command: `cd CS-467-S2018`



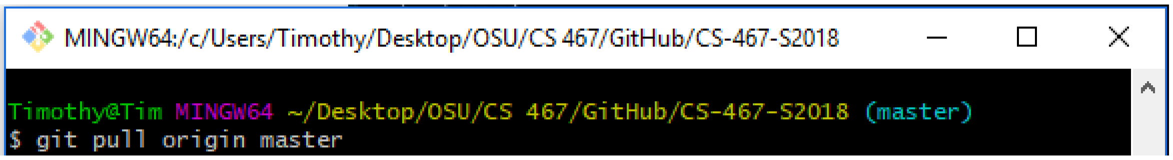
```
MINGW64:/c/Users/Timothy/Desktop/OSU/CS 467/Toss/CS-467-S2018
Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/Toss
$ ls
CS-467-S2018/

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/Toss
$ cd CS-467-S2018

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/Toss/CS-467-S2018 (master)
$ ls
 CS467_AwardHub/     'guides & manuals'/   README.md
'database (.sql)'/   public_html/
```

(alternatively: simply open the “CS-467-S2018” folder in file explorer, right click, and select the “Git Bash Here” option)

Pull Changes to Local Repo

Abstract: This command should be used to pull down any changes made to the GitHub repo by another group member. Though you can use this command as much as you like, it is not that imperative. You will be warned if the repo is not up to date when you try to send a push command. Since we are all working on different files, I don't need to go into the specifics of what happens when two people are working on the same file at one time. If it happens just post a message to the group discussion board.

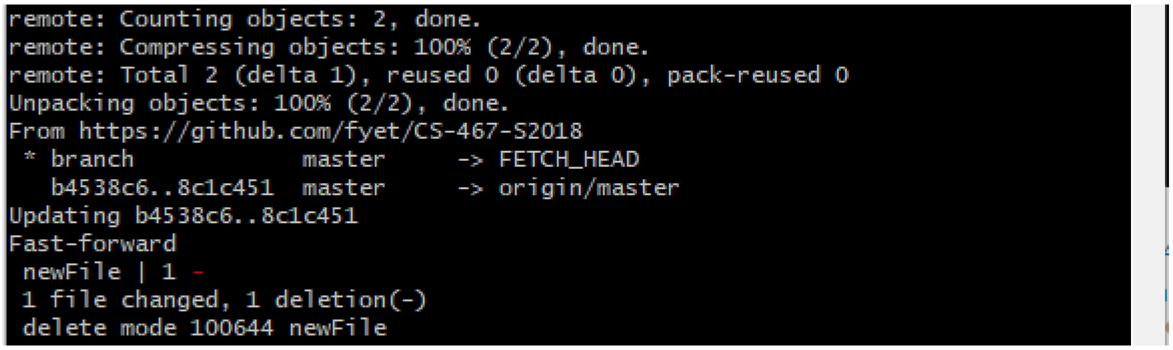
Command: `git pull origin [branch]`



```
MINGW64:/c/Users/Timothy/Desktop/OSU/CS 467/GitHub/CS-467-S2018

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ git pull origin master
```

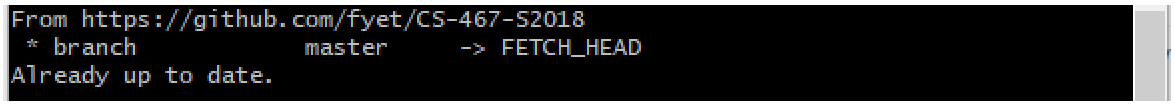
After hitting enter:

1. If changes



```
remote: Counting objects: 2, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 2 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (2/2), done.
From https://github.com/fyet/CS-467-S2018
 * branch            master     -> FETCH_HEAD
   b4538c6..8c1c451  master     -> origin/master
Updating b4538c6..8c1c451
Fast-forward
 newFile | 1 -
 1 file changed, 1 deletion(-)
 delete mode 100644 newFile
```

2. If no changes:



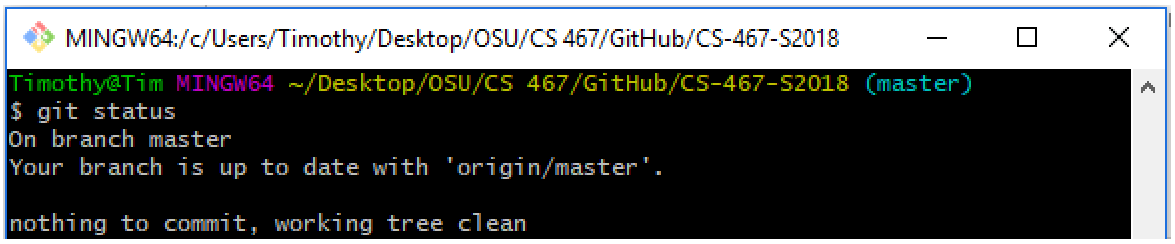
```
From https://github.com/fyet/CS-467-S2018
 * branch            master     -> FETCH_HEAD
Already up to date.
```

Apply Changes to Local Git Repo

Abstract: There are two commands that need to be used to successfully apply changes to your local git repo. They are “add” and “commit”. The “status” command helps us track where in this process we are if we lose track. An example is below

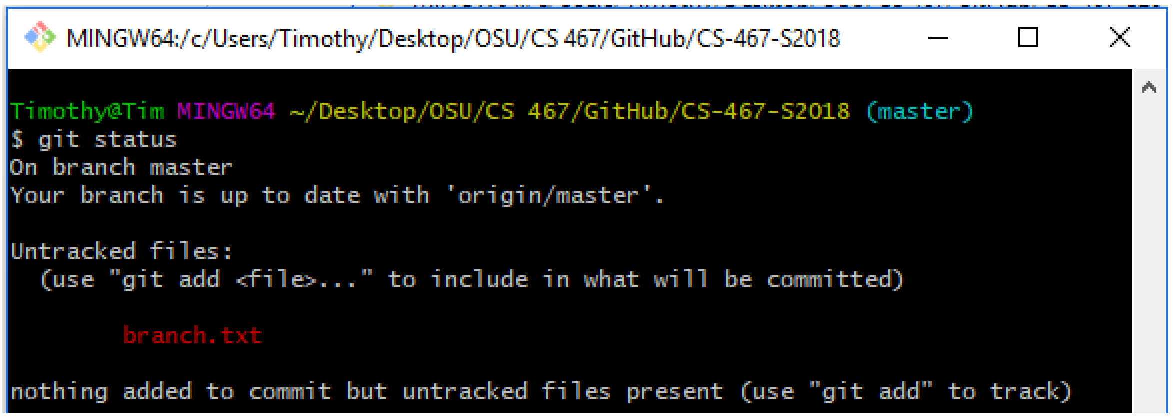
1. Before making any changes, I checked the status. I can see my Git repo is up to date

   Command: `git status`



```
MINGW64:/c/Users/Timothy/Desktop/OSU/CS 467/GitHub/CS-467-S2018
Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
```

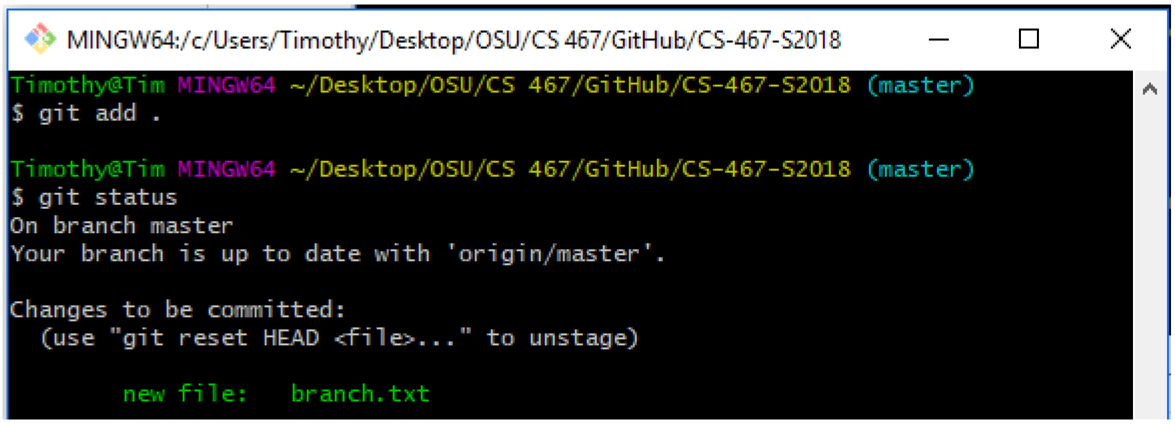
2. I used a text editor to create a new file called “branch.txt”. I saved the file in the “CS-467-S2018” directory. The “status” command now shows I have made untracked changes.



```
MINGW64:/c/Users/Timothy/Desktop/OSU/CS 467/GitHub/CS-467-S2018

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (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)

        branch.txt

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

3. Git doesn’t know which of the untracked changes we want to commit to the branch, which is why we need to use the “add” command to tell it which ones we want to move forward with. For our purposes, we can just add all the files to the stating area by using the dot operator.

   Command: `git add .`

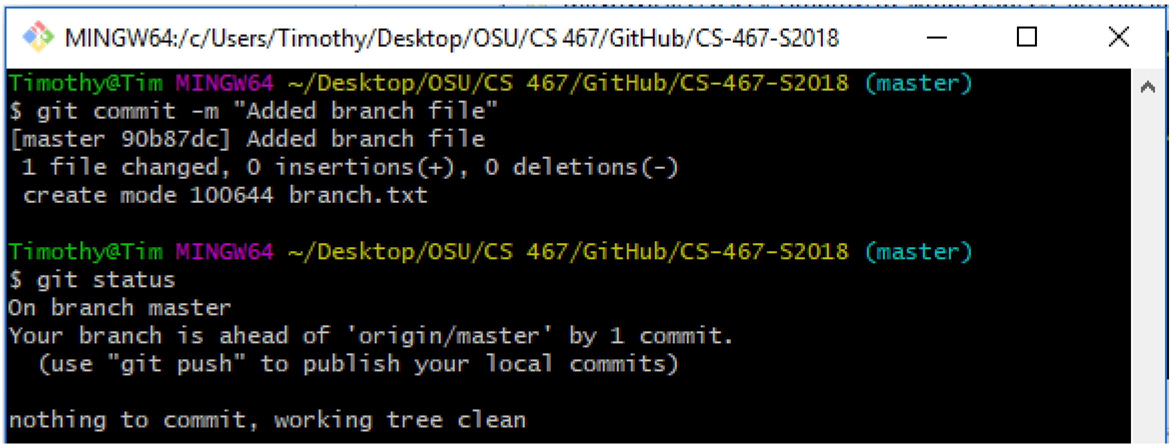


```
MINGW64:/c/Users/Timothy/Desktop/OSU/CS 467/GitHub/CS-467-S2018
Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ git add .

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        new file:   branch.txt
```

(We can now see we have added our changes to the staging area after running the “status” command)

4. Now we need to commit our changes. This “finalizes/records” the changes to the local repo.

Command: `git commit -m "[Description of commits here]"`


MINGW64:/c/Users/Timothy/Desktop/OSU/CS 467/GitHub/CS-467-S2018

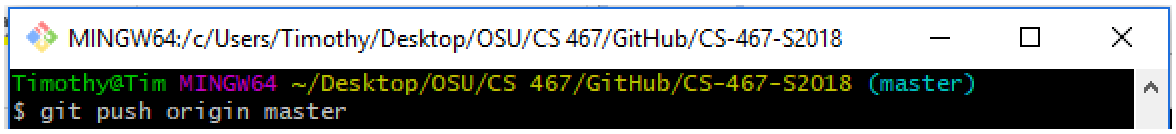
```
Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ git commit -m "Added branch file"
[master 90b87dc] Added branch file
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 branch.txt

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

(Running the “status” command show us that our local repo is once again up to date)

Push Local Changes to GitHub Repo

Abstract: Now that everything is committed to our local Git repo, we can update the public GitHub repo with the "push" command. We will not be able to do this if the steps in the previous section were not followed. Once code is pushed, other group members can then use the aforementioned "pull" command to copy changes to their local repos.

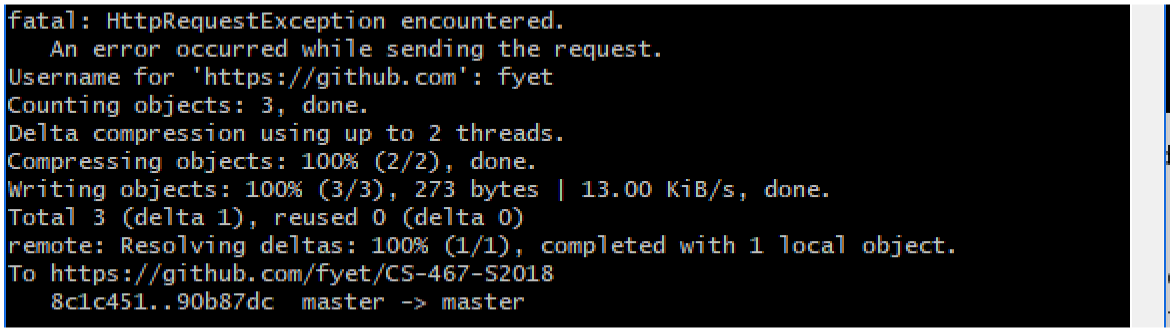
Command: `git push origin [branch]`



MINGW64:/c/Users/Timothy/Desktop/OSU/CS 467/GitHub/CS-467-S2018

```
Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ git push origin master
```

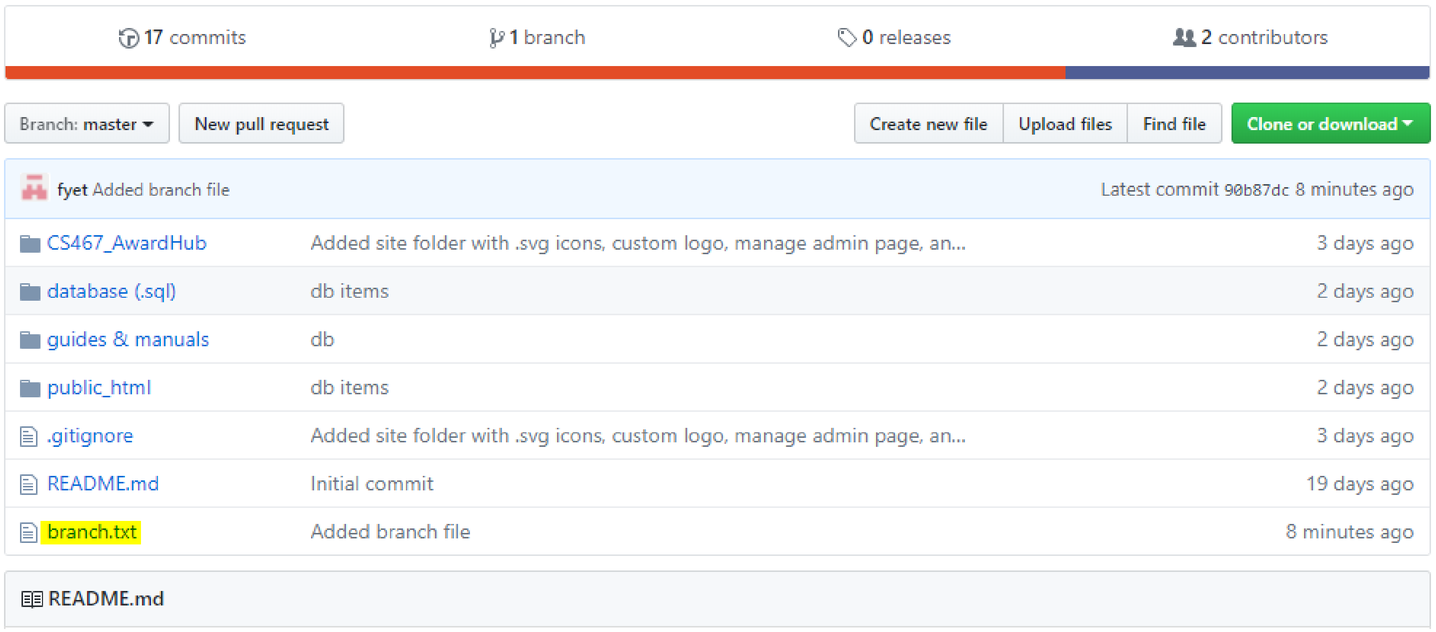
After hitting enter:



```
fatal: HttpRequestException encountered.
   An error occurred while sending the request.
Username for 'https://github.com': fyet
Counting objects: 3, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 273 bytes | 13.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/fyet/CS-467-S2018
   8c1c451..90b87dc  master -> master
```

(Note: I received a "fatal" error because I needed to authenticate. Simply plug in your username if this occurs and a popup for your password will present itself.)

After refreshing GitHub:



fyet Added branch file — Latest commit 90b87dc 8 minutes ago

CS467_AwardHub	Added site folder with .svg icons, custom logo, manage admin page, an...	3 days ago
database (.sql)	db items	2 days ago
guides & manuals	db	2 days ago
public_html	db items	2 days ago
.gitignore	Added site folder with .svg icons, custom logo, manage admin page, an...	3 days ago
README.md	Initial commit	19 days ago
branch.txt	Added branch file	8 minutes ago

README.md

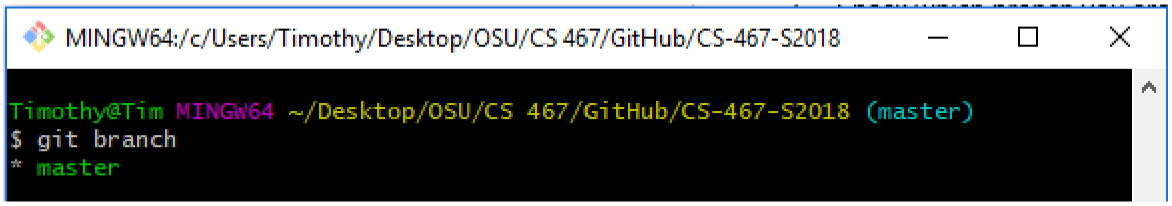
Branch Creation and Navigation

Abstract: I personally recommend we use branches sparingly and simply push all working code to master. We don't want to over complicate things too much for us. The reasons we would want to create a branch are:

- If you are working on two different versions of your code and are not sure which one you want to use
- If you have code that is not working and would like to push it to the repo for the group's help, but pushing this particular code to master would break other finished components of our program this are currently working in master.
- Other preferences the group specify during group chat

1. Check which branch you are currently in / view lists of available branches

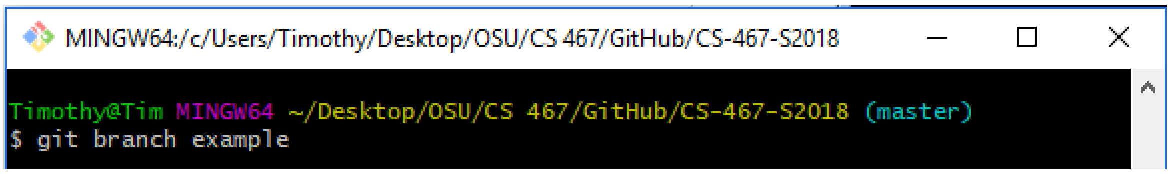
   Command: `git branch`



```
MINGW64:/c/Users/Timothy/Desktop/OSU/CS 467/GitHub/CS-467-S2018

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ git branch
* master
```

2. Create a new branch

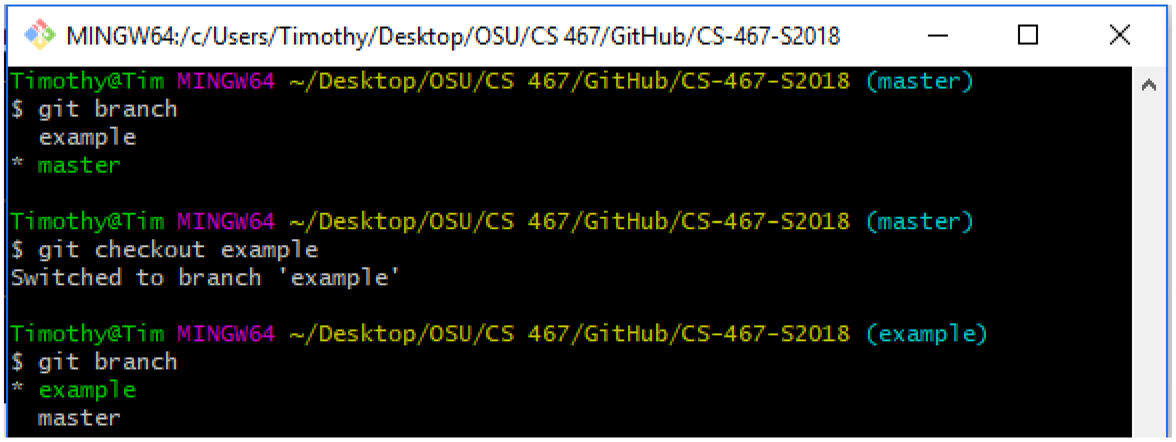
   Command: `git branch [branch name]`



```
MINGW64:/c/Users/Timothy/Desktop/OSU/CS 467/GitHub/CS-467-S2018

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ git branch example
```

3. Navigate to a new branch

   Command: `git checkout [name of branch you would like to go to]`

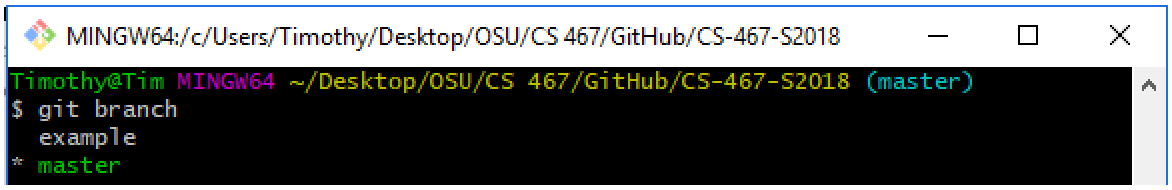


```
MINGW64:/c/Users/Timothy/Desktop/OSU/CS 467/GitHub/CS-467-S2018
Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ git branch
  example
* master

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ git checkout example
Switched to branch 'example'

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (example)
$ git branch
* example
  master
```

Branch Visualization in Local Directory

Abstract: You don't get a new, separate local directory for your branches. When you switch branches, the same directory you are working out of will change behind the scenes. File contents will change from one version to the other. This can be tricky if you lose track of which git branch your directory is pointing to while working through text editors in file explorer. You could end up writing over code you wanted to keep. Here is one trick to help visualize where you are:

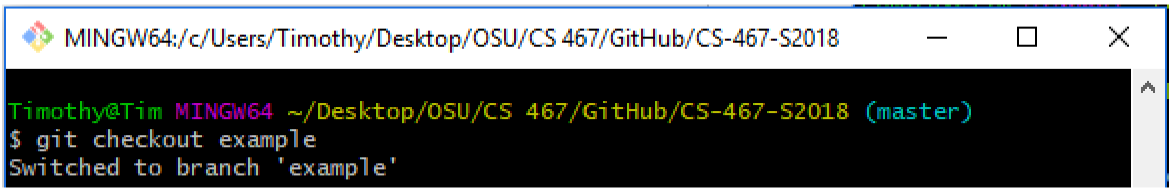
1. I previously created a "branch.txt" document to show how to apply files to a local git and push to GitHub. I can see that I am currently in master (below).



```
MINGW64:/c/Users/Timothy/Desktop/OSU/CS 467/GitHub/CS-467-S2018
Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ git branch
  example
* master
```

2. I edited this document via text editor (you could also just use vim from git bash command line) and entered the contents "Master" in it. I made changes, so I need to run my "add" and "commit" commands.

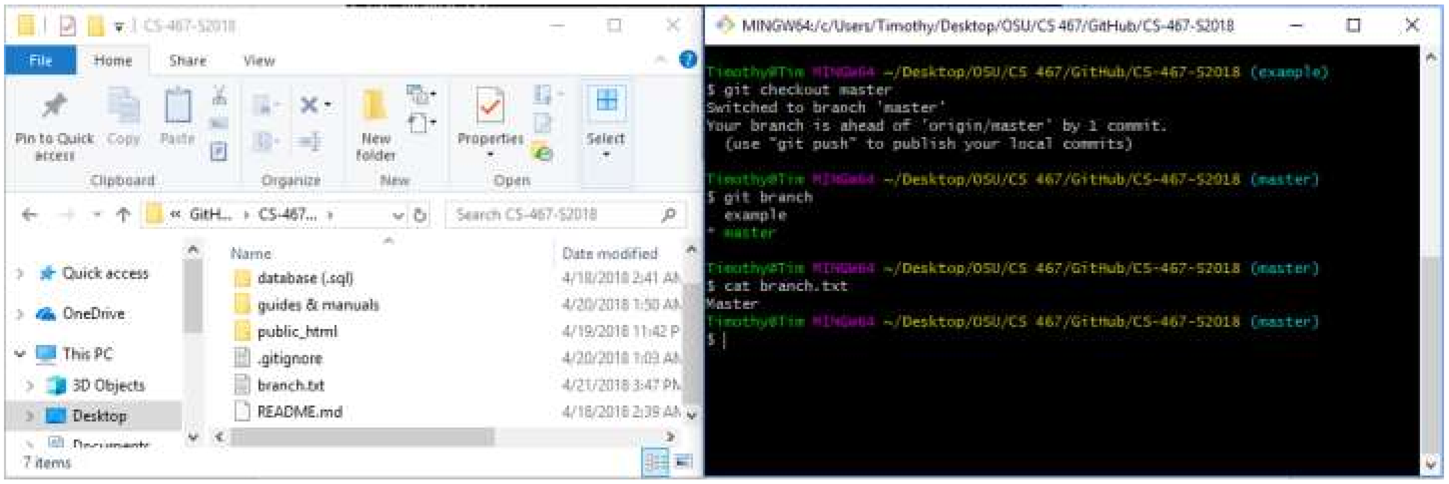
3. I then switched branch and jump over to "Example"



```
MINGW64:/c/Users/Timothy/Desktop/OSU/CS 467/GitHub/CS-467-S2018

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ git checkout example
Switched to branch 'example'
```

4. I edited the "branch.txt" document via text editor and entered the contents "Example" in it. I made changes, so once again I need to run my "add" and "commit" commands.

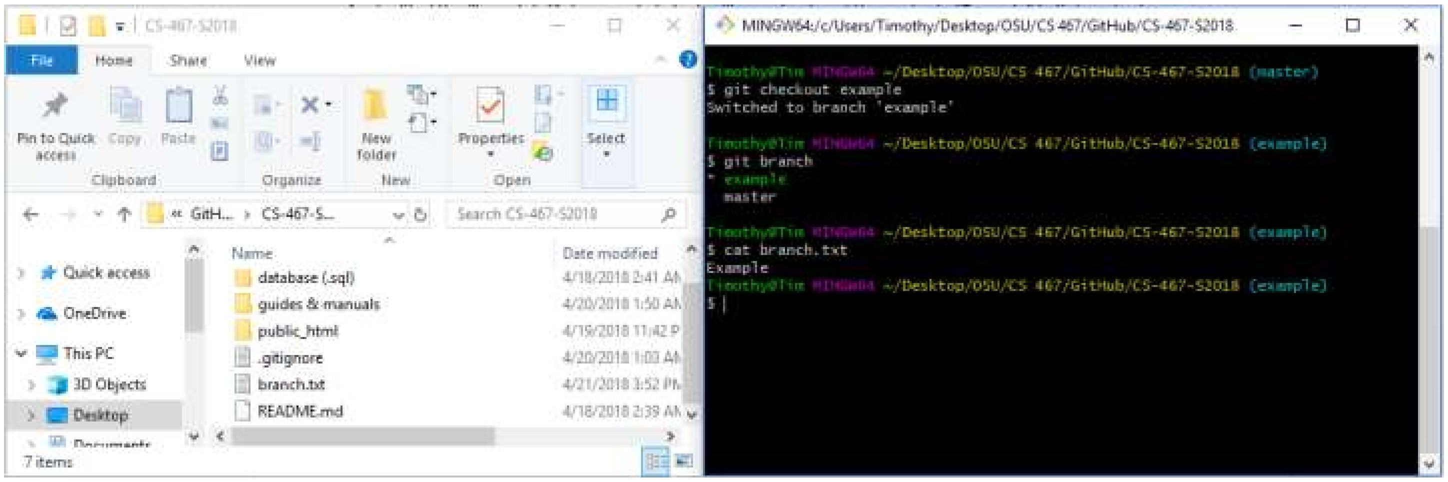
5. Below I have my local directory and a git bash shell up side by side. As you can see, I am on the master branch. When I cat the "branch.txt" (or open it in a text editor from file explorer), it will show "Master" as the file contents.



Name	Date modified
database (.sql)	4/18/2018 2:41 A
guides & manuals	4/20/2018 1:50 A
public_html	4/19/2018 11:42 P
.gitignore	4/20/2018 1:03 A
branch.txt	4/21/2018 3:47 P
README.md	4/16/2018 2:39 A

```
MINGW64:/c/Users/Timothy/Desktop/OSU/CS 467/GitHub/CS-467-S2018
Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (example)
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ git branch
  example
* master

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ cat branch.txt
Master
Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ |
```

6. When I switch over to the “example” branch however, I can open the same “branch.txt” file in the same directory from file explorer. The contents now show “Example” instead of “Master”. I have use the cat command to display the different in contents via the command line.



Name	Date modified
database (.sql)	4/18/2018 2:41 AM
guides & manuals	4/20/2018 1:50 AM
public_html	4/19/2018 11:42 P
.gitignore	4/20/2018 1:03 AM
branch.txt	4/21/2018 3:52 PM
README.md	4/18/2018 2:39 AM

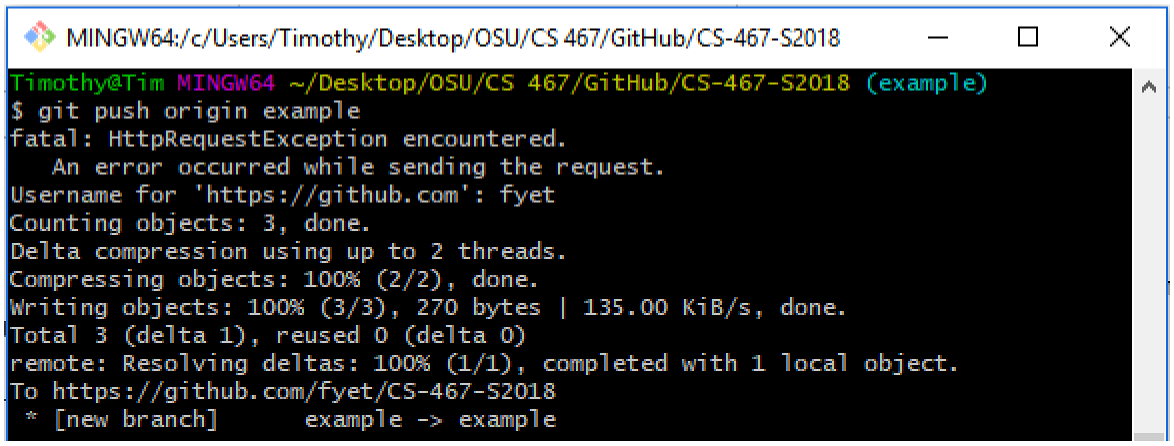
```
MINGW64:/c/Users/Timothy/Desktop/OSU/CS 467/GitHub/CS-467-S2018

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (master)
$ git checkout example
Switched to branch 'example'

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (example)
$ git branch
* example
  master

Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (example)
$ cat branch.txt
Example
Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (example)
$ |
```

Basically it always good to double check which branch you are manipulating in git bash when working out of file explorer, especially before pushing any changes to the public GitHub repo. It can be easy to accidentally start manipulating unintended versions of files.

Pull vs. Pull Request and Merge

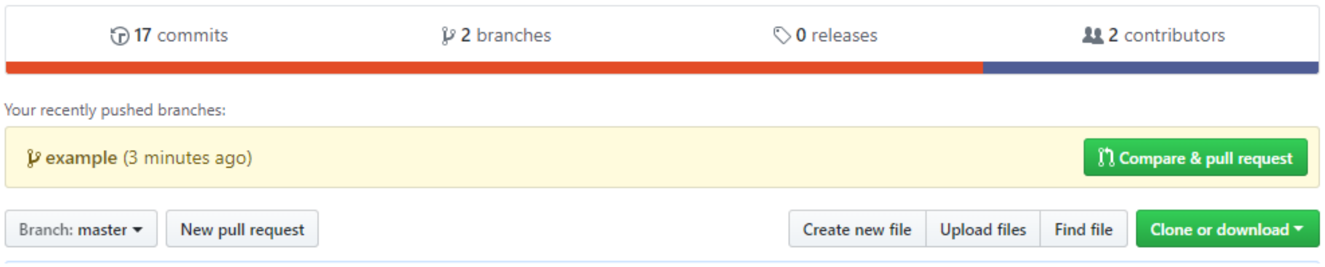
Abstract: A previous section showed how the "pull" command through git bash retrieved code updates from the public GitHub repo. But how do we merge a branch to master? For that we use a pull request, but through GitHub instead of through git bash. I will show an example using the branch I created previously.

1. Start out by pushing my new branch to the GitHub repo (after entering "add" and "commit" to apply any changes I made to local code within the branch).



```
MINGW64:/c/Users/Timothy/Desktop/OSU/CS 467/GitHub/CS-467-S2018
Timothy@Tim MINGW64 ~/Desktop/OSU/CS 467/GitHub/CS-467-S2018 (example)
$ git push origin example
fatal: HttpRequestException encountered.
   An error occurred while sending the request.
Username for 'https://github.com': fyet
Counting objects: 3, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 270 bytes | 135.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/fyet/CS-467-S2018
 * [new branch]      example -> example
```

2. I can now go to GitHub and see the following. Click on the "Compare & pull request"

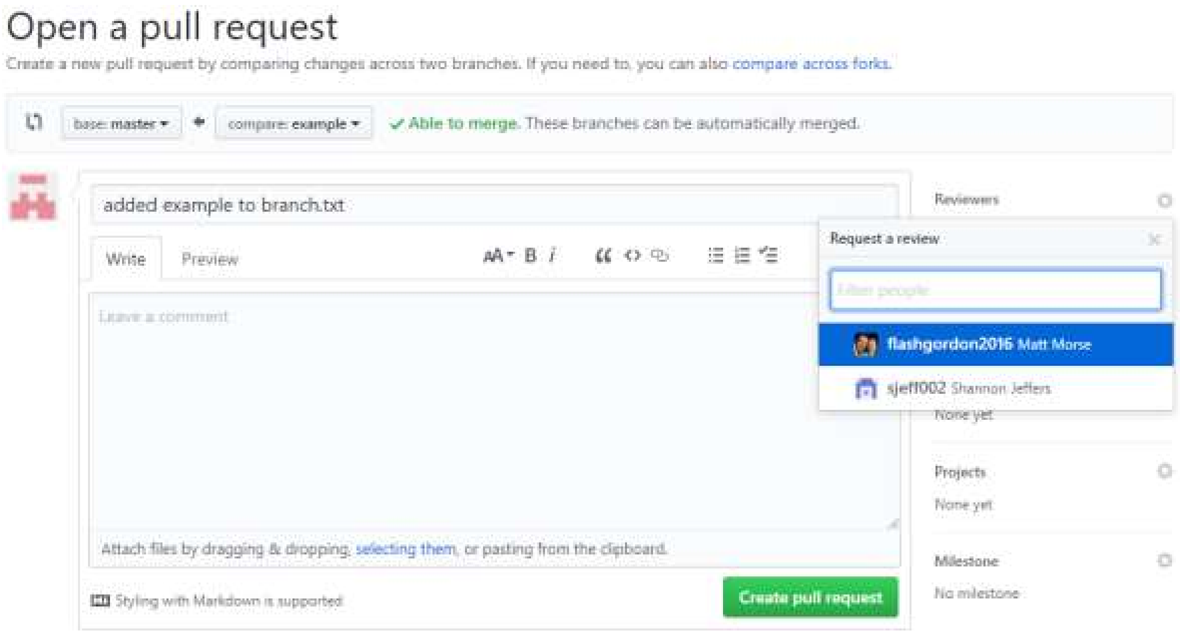


17 commits | 2 branches | 0 releases | 2 contributors

Your recently pushed branches:

example (3 minutes ago)

3. Click on the gear aside "Reviewers" and add a group member or two. Then hit "Create pull request".



Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.

base: master ← compare: example ✓ Able to merge. These branches can be automatically merged.

added example to branch.txt

Leave a comment

Attach files by dragging & dropping, selecting them, or pasting from the clipboard.

Styling with Markdown is supported

Reviewers

Request a review

Filter people

flashgordon2016 Matt Morse

sjeff002 Shannon Jeffers

None yet

Projects

None yet

Milestone

No milestone

4. The group members can merge these together (I recommend having someone else review and perform the merge just for good measure).