Gitflow Workflow: Visualised

I found Gitflow workflow confusing and I messed up several times, so here's a breakdown of what I've learnt, with screenshots!

It'll also be a useful reference for whenever you take an extended break from git.

THE CONCEPT

Gitflow workflow offers an additional layer of security over **feature branch workflow**.

The two are essentially the same, with one key difference: Gitflow workflow introduces an additional layer (the develop branch) between a given collaborator's feature branch and the master branch.

It can be visualised like this:

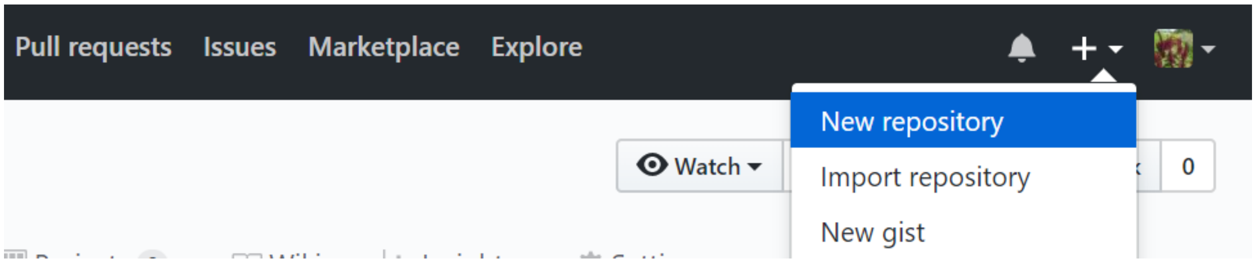
```
MASTER BRANCH
        DEVELOP BRANCH
                CHRISTY'S FEATURE 01
                CHRISTY'S FEATURE 02
                DELLE'S FEATURE 01
```

Master is reserved for the deployment of new versions. All development work is, as the name suggests, contained within develop branch.

An individual collaborator would work on their feature locally. When they are ready to commit a good, clean module or aspect (this follows good design principles!) of their feature, they stage, commit, and push their feature branch only to the remote repo.

THE PROCESS

CREATE NEW REPO ON GITHUB.



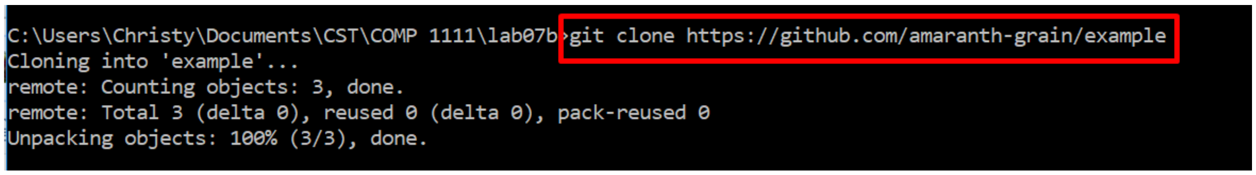
(TIP: You can rename your repo through "Settings" at the top if you make a typo.)

SEND INVITE TO COLLABORATORS.

Settings > Collaborators > Find them on Github

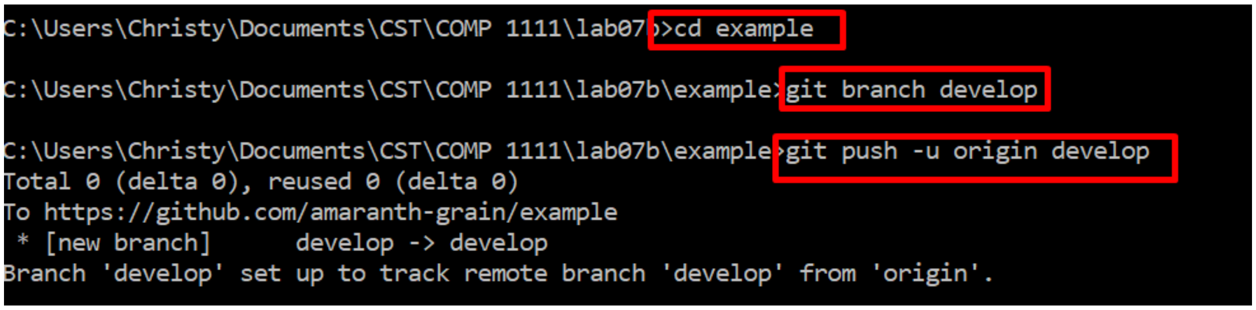
(TIP: Make sure your fellow collaborators accept your invitation! If they do not, they do not have the necessary permissions to contribute to your project.)

CLONE REMOTE REPO TO LOCAL.



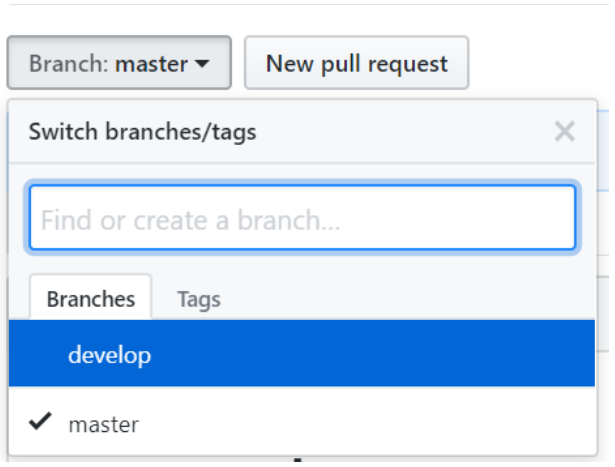
```
C:\Users\Christy\Documents\CST\COMP 1111\lab07b>git clone https://github.com/amaranth-grain/example
Cloning into 'example'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
```

CD INTO THE LOCAL REPO
CREATE DEVELOP BRANCH FROM MASTER
PUSH IT TO REMOTE REPO

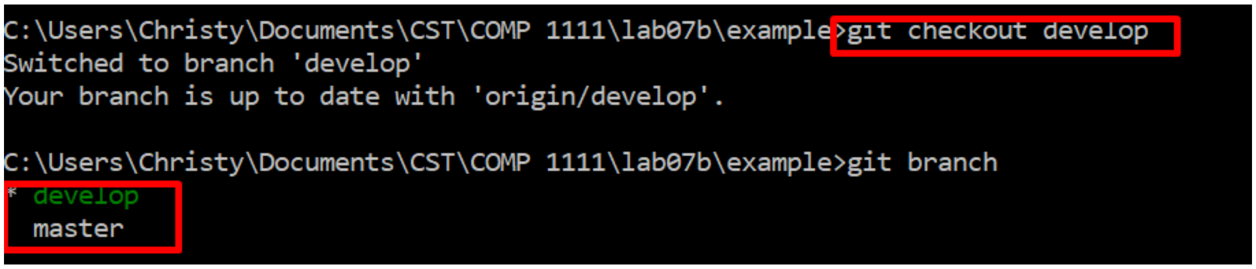


```
C:\Users\Christy\Documents\CST\COMP 1111\lab07b>cd example

C:\Users\Christy\Documents\CST\COMP 1111\lab07b\example>git branch develop

C:\Users\Christy\Documents\CST\COMP 1111\lab07b\example>git push -u origin develop
Total 0 (delta 0), reused 0 (delta 0)
To https://github.com/amaranth-grain/example
 * [new branch]      develop -> develop
Branch 'develop' set up to track remote branch 'develop' from 'origin'.
```

If you look on Github, you should now see the develop branch.



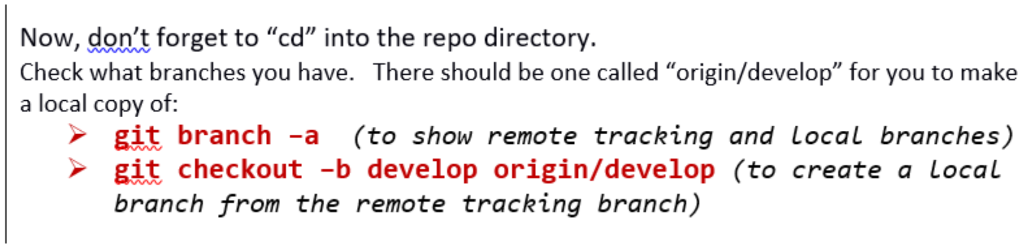
If you are the **one who created the remote repo**, DO NOT clone the repo again like the lab instructions say. You already have the develop branch (locally and remotely). All you need to do is checkout develop.



```
C:\Users\Christy\Documents\CST\COMP 1111\lab07b\example>git checkout develop
Switched to branch 'develop'
Your branch is up to date with 'origin/develop'.

C:\Users\Christy\Documents\CST\COMP 1111\lab07b\example>git branch
* develop
  master
```

~~~~~~~~~~~~~~~~~~~~~

If you are a collaborator who has been invited to join the project, this is when you **git clone <url>** so you have access to the develop branch, and follow these instructions:



Now, don't forget to "cd" into the repo directory.
Check what branches you have. There should be one called "origin/develop" for you to make a local copy of:

- `git branch -a` *(to show remote tracking and local branches)*
- `git checkout -b develop origin/develop` *(to create a local branch from the remote tracking branch)*

Figure 1 Instructions from Carly's lab.

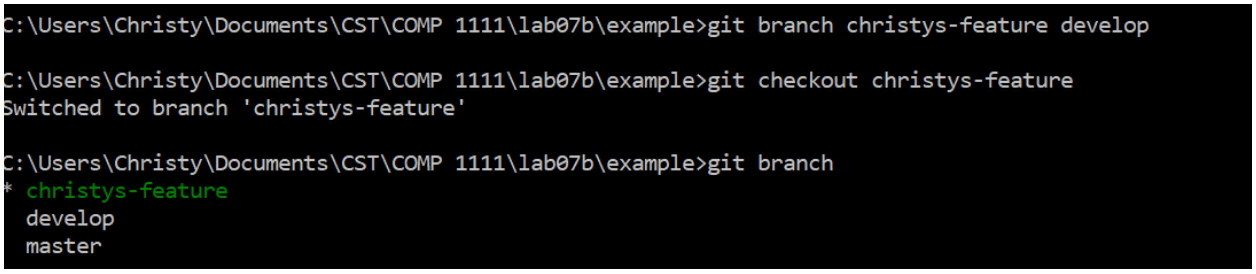
Once you complete these steps, you will be within the develop branch. That is, you're caught up with the owner of the repo.

===

Everyone should be in the **develop branch**.

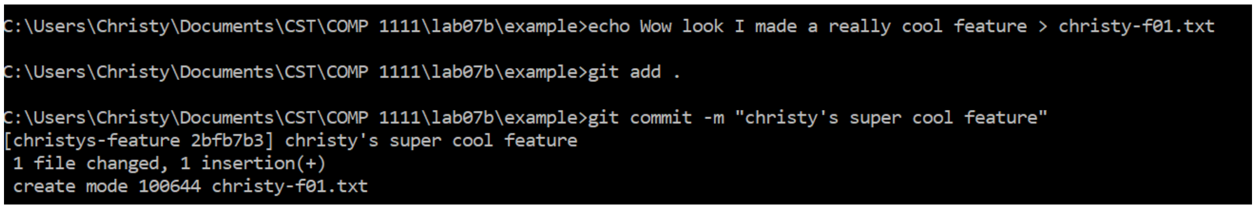
*(TIP: You can check by typing **git branch**.)*

LOCALLY CREATE FEATURE BRANCH
CHECKOUT YOUR FEATURE BRANCH



```
C:\Users\Christy\Documents\CST\COMP 1111\lab07b\example>git branch christys-feature develop

C:\Users\Christy\Documents\CST\COMP 1111\lab07b\example>git checkout christys-feature
Switched to branch 'christys-feature'

C:\Users\Christy\Documents\CST\COMP 1111\lab07b\example>git branch
* christys-feature
  develop
  master
```
~~~~~~~~~~~~~~~~~~~~~

WORK ON YOUR FEATURE.
STAGE AND COMMIT WHEN READY.

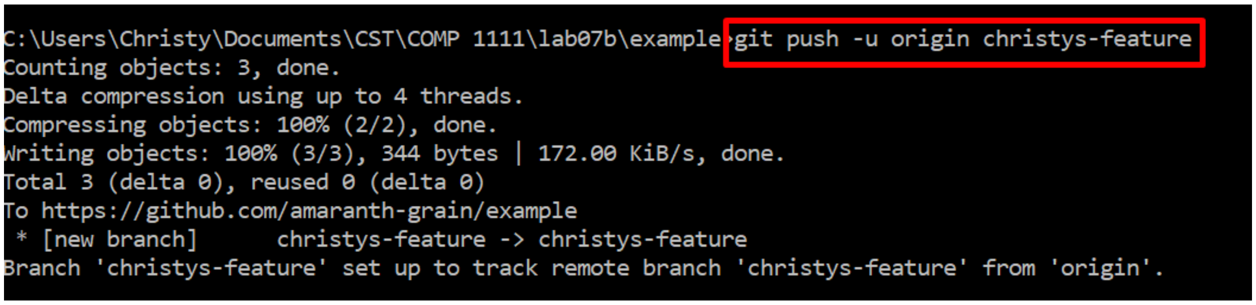


```
C:\Users\Christy\Documents\CST\COMP 1111\lab07b\example>echo Wow look I made a really cool feature > christy-f01.txt

C:\Users\Christy\Documents\CST\COMP 1111\lab07b\example>git add .

C:\Users\Christy\Documents\CST\COMP 1111\lab07b\example>git commit -m "christy's super cool feature"
[christys-feature 2bfb7b3] christy's super cool feature
 1 file changed, 1 insertion(+)
 create mode 100644 christy-f01.txt
```

===

PUSH YOUR FEATURE BRANCH ONLY.

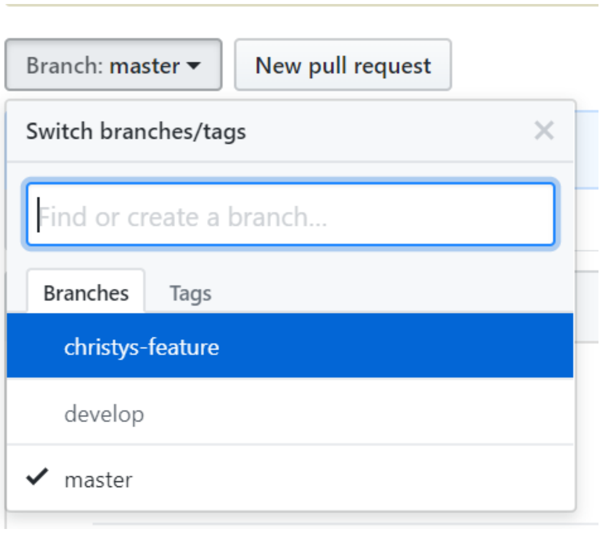
//This is the pull request method; the merge method (a) makes your contributions less transparent and (b) is easy enough to follow along with Carly’s lab07 notes.



```
C:\Users\Christy\Documents\CST\COMP 1111\lab07b\example>git push -u origin christys-feature
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 344 bytes | 172.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/amaranth-grain/example
 * [new branch]      christys-feature -> christys-feature
Branch 'christys-feature' set up to track remote branch 'christys-feature' from 'origin'.
```

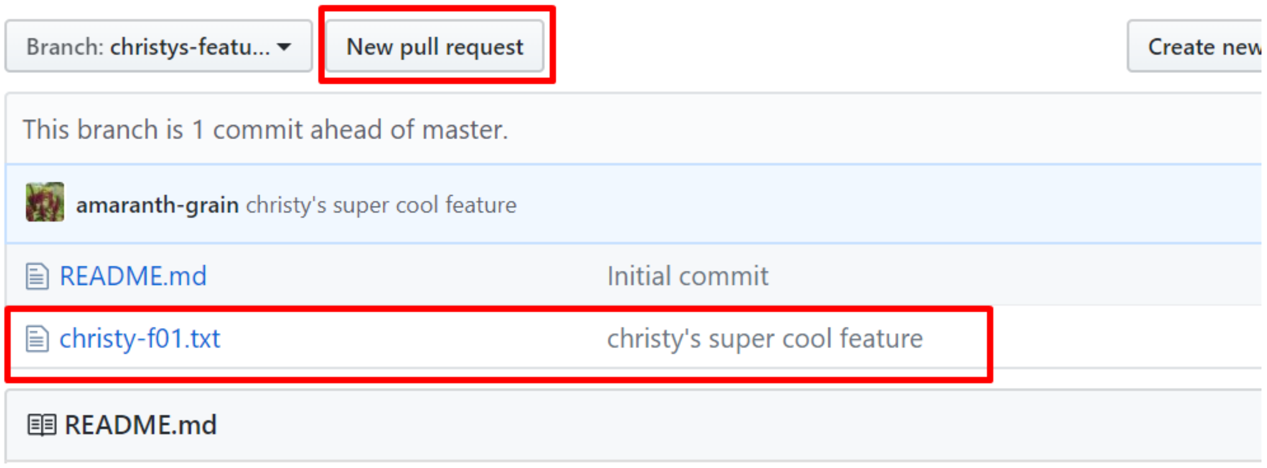
GO TO GITHUB. YOUR FEATURE BRANCH SHOULD SHOW UP.

SELECT IT.



==

YOUR FEATURE WILL APPEAR ONLY IN THIS FEATURE BRANCH.
IF YOU LOOK AT DEVELOP OR MASTER, IT WILL NOT BE THERE (YET).
CLICK THE "New pull request" BUTTON.



This branch is 1 commit ahead of master.

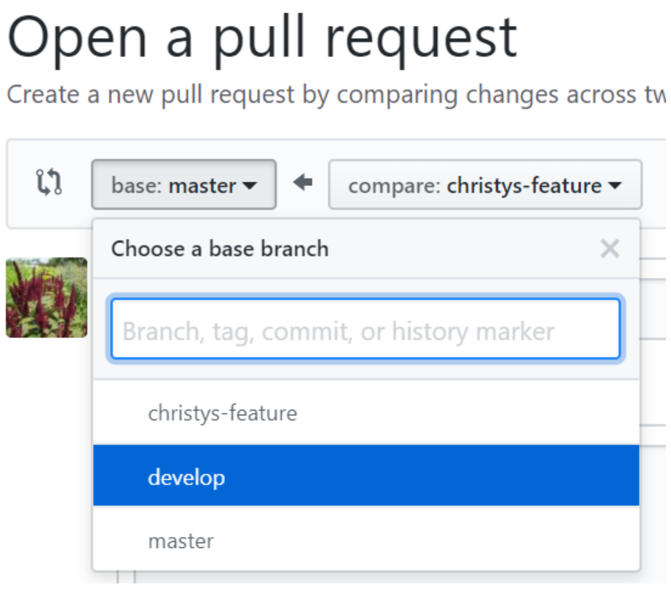
amaranth-grain christy's super cool feature

README.md	Initial commit
christy-f01.txt	christy's super cool feature

README.md

OPEN THE DROPDOWN MENU AND SELECT 'Develop'.

(TIP: Consider the hierarchy of the gitflow workflow. You're in the feature branch. The next level up is the develop branch. You want to leave master alone for now.)

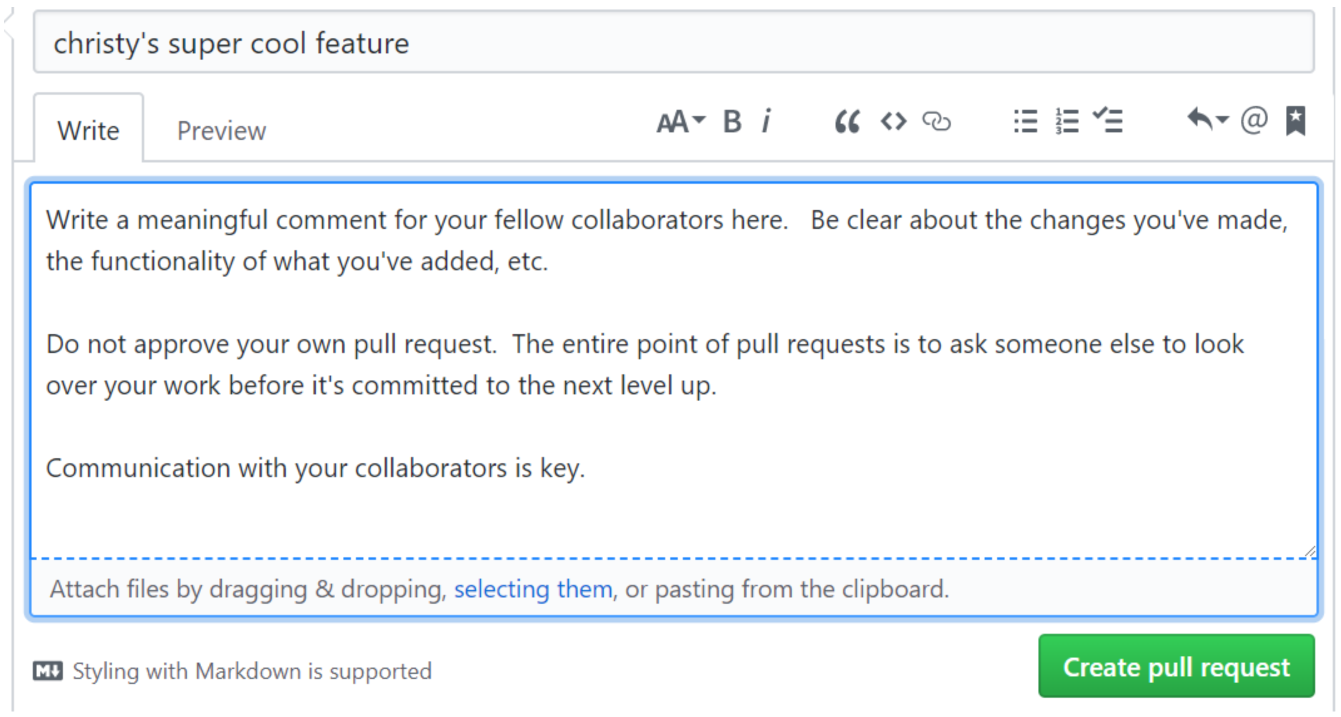


Open a pull request

Create a new pull request by comparing changes across tw

base: master
compare: christys-feature
Choose a base branch
Branch, tag, commit, or history marker
christys-feature
develop
master

CLICK "Create pull request" AFTER WRITING A COMMENT.


christy's super cool feature

Write a meaningful comment for your fellow collaborators here. Be clear about the changes you've made, the functionality of what you've added, etc.

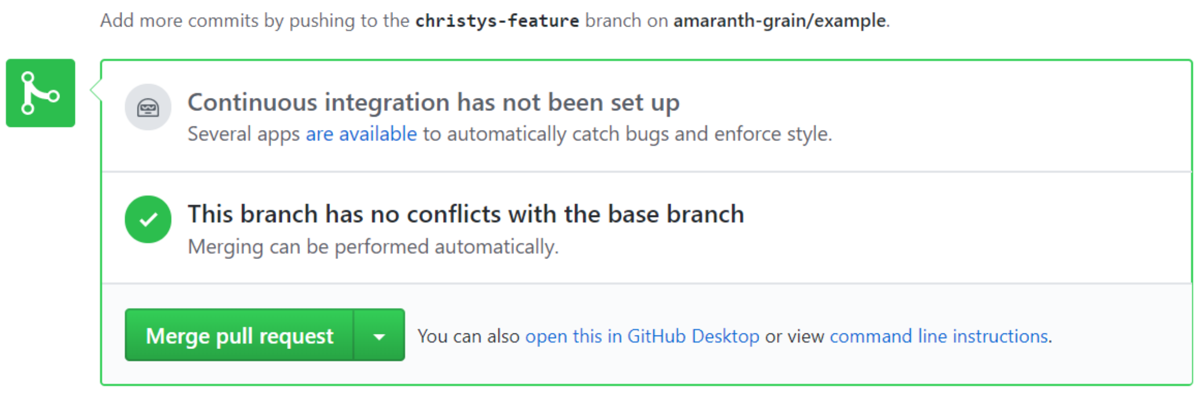
Do not approve your own pull request. The entire point of pull requests is to ask someone else to look over your work before it's committed to the next level up.

Communication with your collaborators is key.

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

Someone else should go to the Pull Requests tab from the top, read your comments, hash out any conflicts with you, and then merge the request.

(TIP: Remember to click twice – you must confirm the merge.)


Add more commits by pushing to the **christys-feature** branch on **amaranth-grain/example**.

Continuous integration has not been set up
Several apps are available to automatically catch bugs and enforce style.

This branch has no conflicts with the base branch
Merging can be performed automatically.

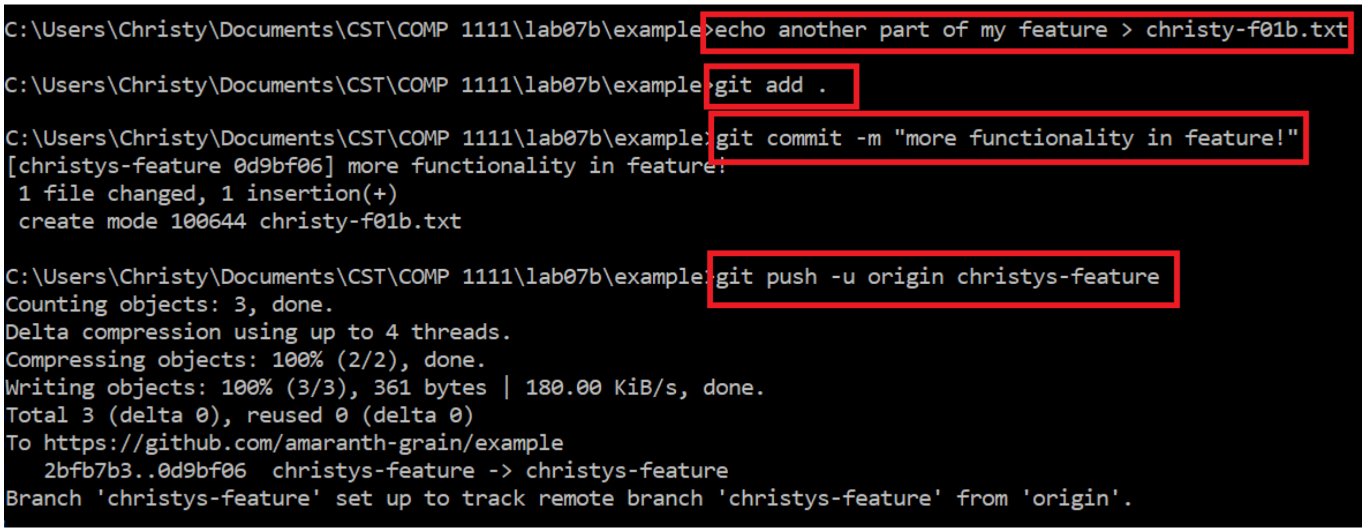
You can also open this in GitHub Desktop or view command line instructions.

If you go back to your repo, you will see that your feature has been added to the develop branch, but not the master branch.

If you were working on a project, you would continue to work on your feature, push your feature branch to the remote repo, initiate a pull request in GitHub, and get someone else to merge your feature branch to develop.

I'll add another text file from command line to represent another aspect of the "software feature" that I'm working on.

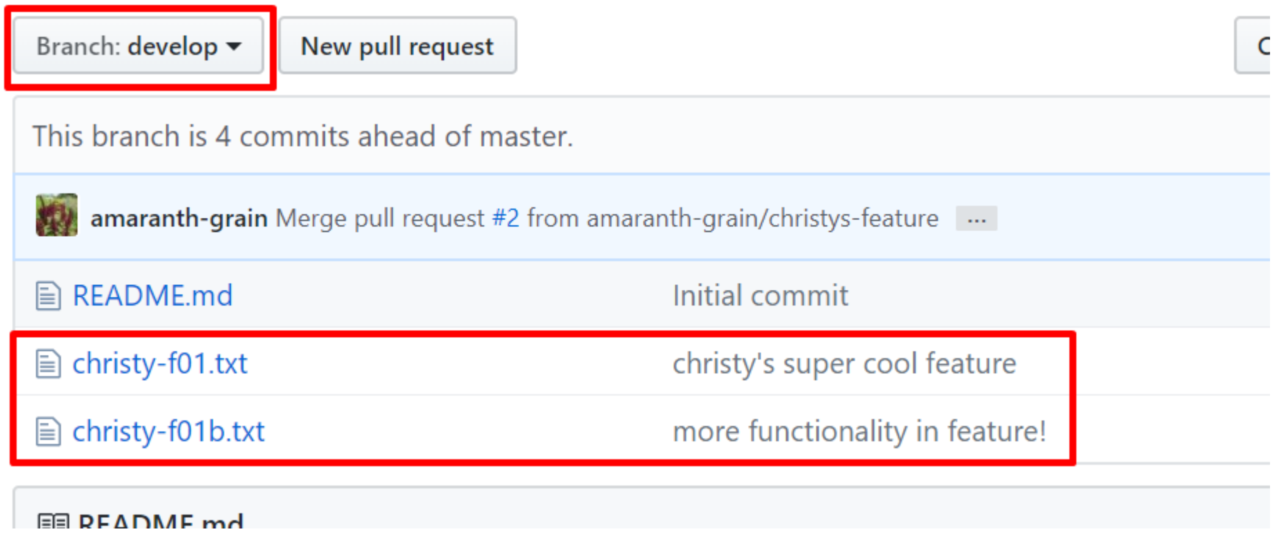
*(TIP: When in doubt, type **git status** to see what's happening – like to see if you've remembered to commit. Type **git branch -a** if you don't know where you are and want to see all branches, including remote ones.)*



```
C:\Users\Christy\Documents\CST\COMP 1111\lab07b\example>echo another part of my feature > christy-f01b.txt

C:\Users\Christy\Documents\CST\COMP 1111\lab07b\example>git add .

C:\Users\Christy\Documents\CST\COMP 1111\lab07b\example>git commit -m "more functionality in feature!"
[christys-feature 0d9bf06] more functionality in feature!
 1 file changed, 1 insertion(+)
 create mode 100644 christy-f01b.txt

C:\Users\Christy\Documents\CST\COMP 1111\lab07b\example>git push -u origin christys-feature
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 361 bytes | 180.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/amaranth-grain/example
   2bfb7b3..0d9bf06  christys-feature -> christys-feature
Branch 'christys-feature' set up to track remote branch 'christys-feature' from 'origin'.
```

I'll initiate a pull request from my feature branch on GitHub, remembering to change the dropdown menu selection to "develop", because I don't want to touch my master branch.

Someone else will look over my contribution, and merge it to the develop branch when all conflicts have been resolved.



This branch is 4 commits ahead of master.

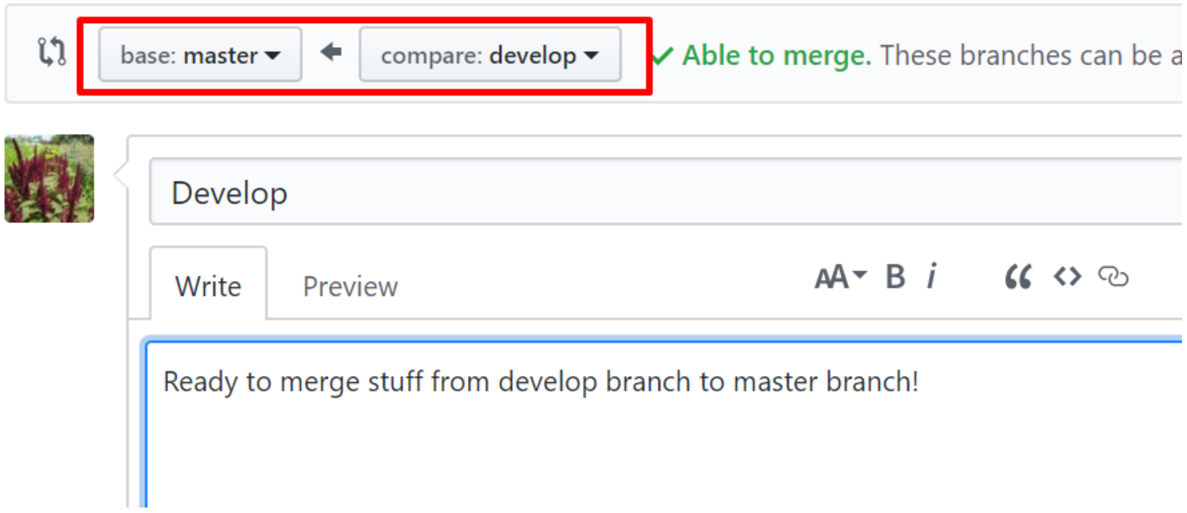
amaranth-grain Merge pull request #2 from amaranth-grain/christys-feature

README.md	Initial commit
christy-f01.txt	christy's super cool feature
christy-f01b.txt	more functionality in feature!

README.md

Wow, my feature is completely ready! It's been tested extensively… time to make a pull request from the **develop branch** to the **master branch**.

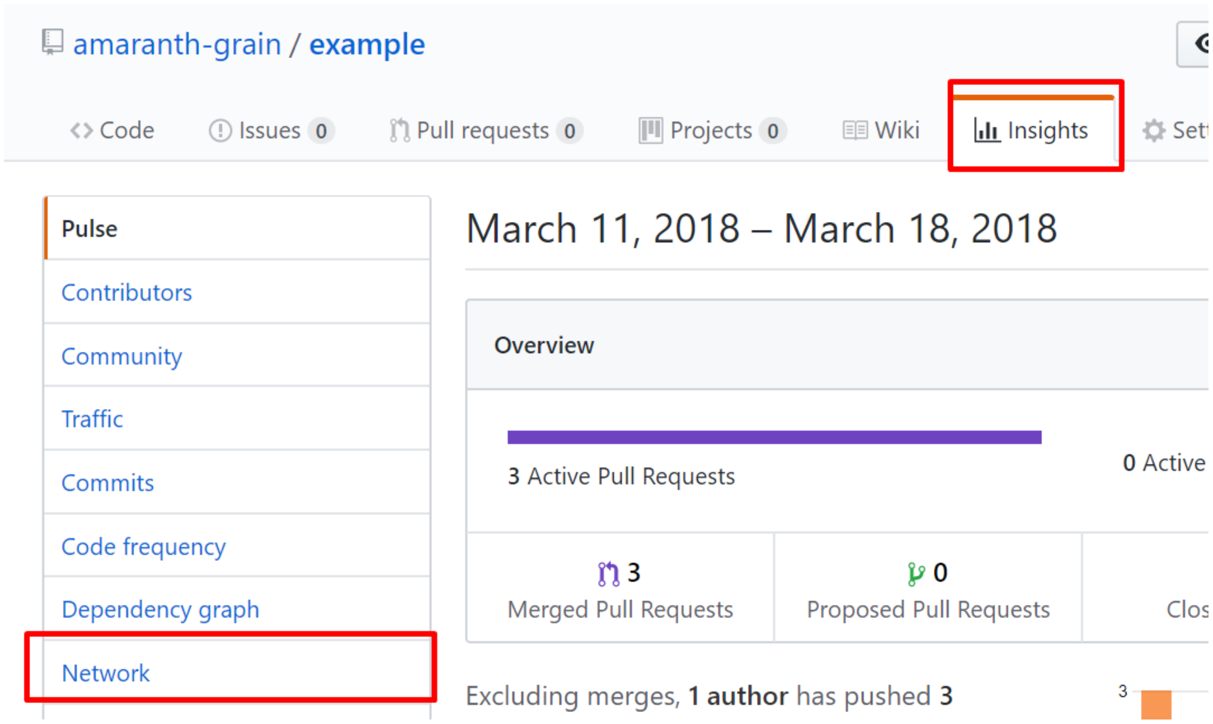
You do exactly the same thing as before – go to develop, click "New pull request", and send a pull request so you can merge stuff from develop to master.



Develop

Ready to merge stuff from develop branch to master branch!

Now for the moment of truth! If you go to Insights > Network, you'll see if you've done everything right.



March 11, 2018 – March 18, 2018

Overview

3 Active Pull Requests

0 Active

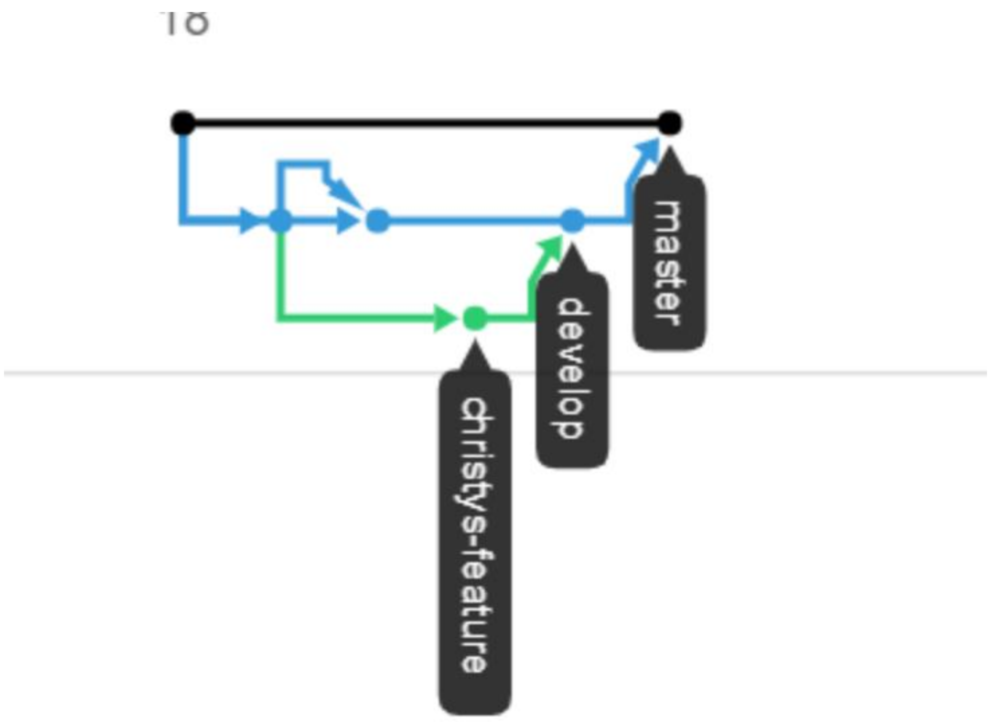
3 Merged Pull Requests

0 Proposed Pull Requests

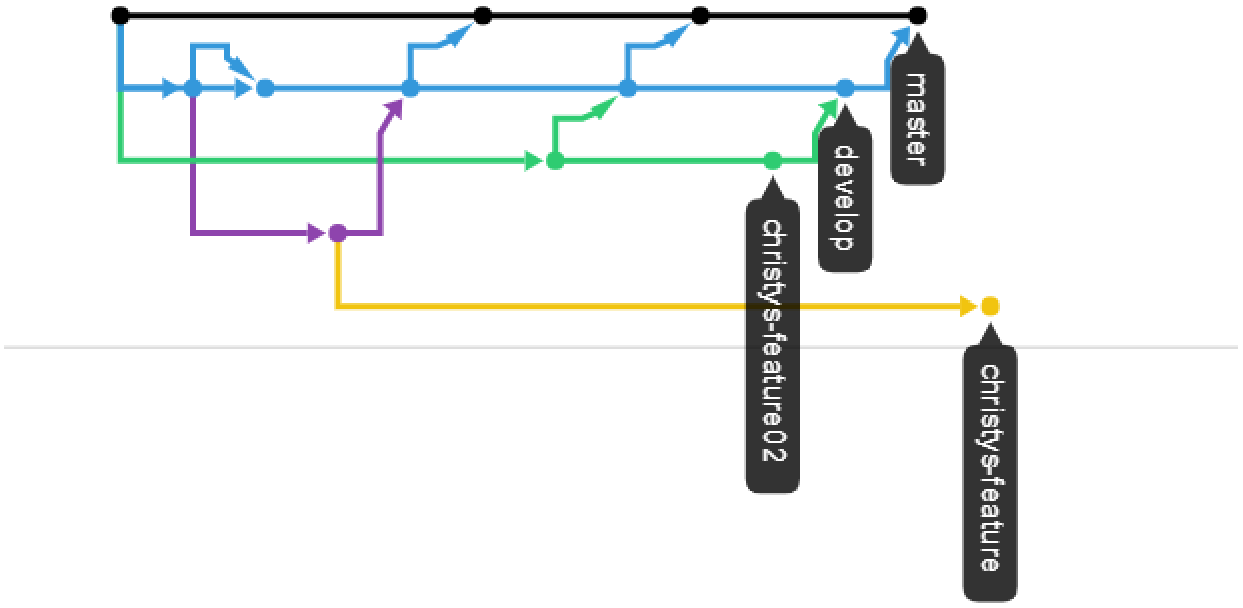
Clos

Excluding merges, **1 author** has pushed **3**

Tada!



If you repeat this process a few more times, it should look something like this:



Notice that master (the black line) and develop (the blue one) are permanent branches. The feature branches are temporary and can be safely deleted once they've been merged into develop.