

Object-Oriented Programming (Python) Tutorial

How to git

Dr. Bianca Schoen-Phelan

bianca.phelan@tudublin.ie

www.biancaphelan.ie



@BSPhelan

KE-3-005a

Objectives

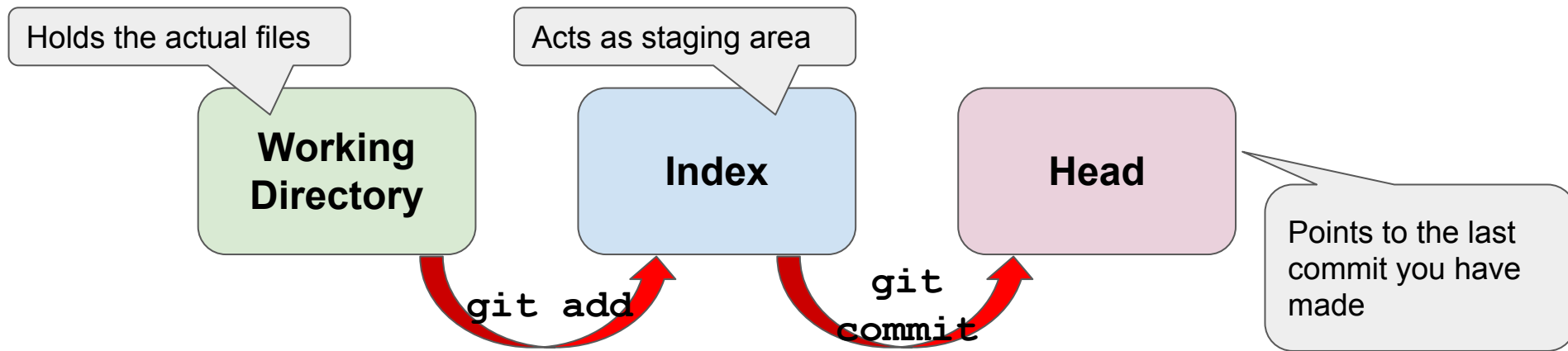
- Create a local repository
- Add local repository to git control and work with files
- Create a local repo for remote git location
- Work with files

Prerequisites

1. You have `git` installed on your local computer
2. You have created a `GitHub` account

git Structure - **important** to understand

- The `git` workflow relies on the following structure



- The `git` repository relies on three trees that `git` maintains, the Working Directory, the Index or stage and the Head

Local Command line `git`

Local command line git Scenario

1. Create a directory you want to work in
2. Create a Python file
3. Make everything `git` controlled
4. Create a new branch and use that branch
 - a. Make changes to the file
 - b. Commit changes
 - c. Revert to a previous version
5. Merge feature branch with master
6. Delete feature branch

Commands

```
0:$ mkdir Project1.0
```

```
1:$ cd Project1.0/
```

```
2:$ vi hello_world.py
```

```
3:$ git init
```

Initialized empty Git repository in

/Users/bianca.schoenphelan/Documents/gitExamples/Project1.0/.git/

```
4:$ git add hello_world.py
```

```
5:$ git status
```

On branch master

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: hello_world.py

Create the directory and files locally, then tell git that this is a git repo using git init.

- Add your file to the git index
- Run a git status to inspect what git thinks about your repository's state of health

Commands cont'd

Attaches your file to the git head

```
6:$ git commit -m "Original File"
```

```
[master (root-commit) ace9ada] Original File
Committer: Bianca SchoenPhelan <bianca.schoenphelan@soc-mbp13-bsp.lan>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:
```

```
git config --global --edit
```

After doing this, you may fix the identity used for this commit with:

```
git commit --amend --reset-author
```

Keep an eye on the history of events.

```
1 file changed, 2 insertions(+)
create mode 100644 hello_world.py
```

```
7:$ git log --oneline
```

```
ace9ada (HEAD -> master) Original File
```


Commands cont'd

```
8:$ git checkout -b feature_1
```

Switched to a new branch 'feature_1'

```
9:$ git branch
```

```
* feature_1  
master
```

```
10:$ cat hello_world.py
```

```
print('Hello world')
```

```
11:$ vi hello_world.py
```

```
12:$ git add hello_world.py
```

Create a new branch off of master and use it immediately

Check with branch we are in

This is our file. We haven't done anything with it yet.

- We are making a change on the file in the vi editor and then
- add the file to the index.
- Don't forget to add a file to the index, you'll get an error message if you try to commit without.!

Commands cont'd

```
13:$ git commit -m "First change in feature_1"
```

```
[feature_1 22c5d4b] First change in feature_1
```

```
Committer: Bianca SchoenPhelan <bianca.schoenphelan@soc-mbp13-bsp.lan>
```

Your name and email address were configured automatically based on your username and hostname. Please check that they are accurate. You can suppress this message by setting them explicitly. Run the following command and follow the instructions in your editor to edit your configuration file:

```
git config --global --edit
```

After doing this, you may fix the identity used for this commit with:

```
git commit --amend --reset-author
```

```
1 file changed, 1 insertion(+)
```

```
14:$ git log --oneline
```

```
22c5d4b (HEAD -> feature_1) First change in feature_1
```

```
ace9ada (master) Original File
```

Committing the change to the git head and then checking the history log

Commands cont'd

```
15:$ vi hello_world.py
```

```
16:$ git add hello_world.py
```

```
17:$ git commit -m "Second change in feature_1"
```

```
[feature_1 198a797] Second change in feature_1
```

```
Committer: Bianca SchoenPhelan <bianca.schoenphelan@soc-mbp13-bsp.lan>
```

Your name and email address were configured automatically based on your username and hostname. Please check that they are accurate. You can suppress this message by setting them explicitly. Run the following command and follow the instructions in your editor to edit your configuration file:

```
git config --global --edit
```

After doing this, you may fix the identity used for this commit with:

```
git commit --amend --reset-author
```

```
1 file changed, 1 insertion(+)
```

```
18:$ git log --oneline
```

```
198a797 (HEAD -> feature_1) Second change in feature_1
```

```
22c5d4b First change in feature_1
```

```
ace9ada (master) Original File
```

Making more changes and committing them.

Commands cont'd

```
19:$ cat hello_world.py
```

```
print('Hello world')
print('First change')
print('Second change')
```

```
20:$ git checkout 22c5d4b
```

Note: checking out '22c5d4b'.

- We look at a previous version.
- This detaches the head
- Do not leave the repo like that! The garbage collector will clean up detached heads.

You are in 'detached HEAD' state. You can look around, make experimental changes and commit them, and you can discard any commits you make in this state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may do so (now or later) by using `-b` with the checkout command again. Example:

```
git checkout -b <new-branch-name>
```

HEAD is now at 22c5d4b First change in feature_1

```
21:$ cat hello_world.py
```

```
print('Hello world')
print('First change')
```

Commands cont'd

Shows you that we are working on a detached head.

```
22:$ git branch
```

```
* (HEAD detached at 22c5d4b)
  feature_1
  master
```

```
23:$ git log --oneline
```

```
22c5d4b (HEAD) First change in feature_1
ace9ada (master) Original File
```

Get back to a workable state.

```
24:$ git checkout feature_1
```

```
Previous HEAD position was 22c5d4b First change in feature_1
Switched to branch 'feature_1'
```

```
25:$ git log --oneline
```

```
198a797 (HEAD -> feature_1) Second change in feature_1
22c5d4b First change in feature_1
ace9ada (master) Original File
```

```
26:$ cat hello_world.py
```

```
print('Hello world')
print('First change')
print('Second change')
```

Command's cont

We didn't like the most recent change and revert the last commit.

```
27:$ git revert HEAD
```

```
[feature_1 d899b2b] Revert "Second change in feature_1"
Committer: Bianca SchoenPhelan <bianca.schoenphelan@soc-mbp13-bsp.lan>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:
```

```
git config --global --edit
```

After doing this, you may fix the identity used for this commit with:

```
git commit --amend --reset-author
```

The history reflects what we have done.

```
1 file changed, 1 deletion(-)
```

```
28:$ git log --oneline
```

```
d899b2b (HEAD -> feature_1) Revert "Second change in feature_1"
198a797 Second change in feature_1
22c5d4b First change in feature_1
ace9ada (master) Original File
```

Commands cont'd

```
29:$ cat hello_world.py
```

```
print('Hello world')
print('First change')
```

```
30:$ git branch
```

```
* feature_1
  master
```

```
31:$ git checkout master
```

```
Switched to branch 'master'
```

```
32:$ git branch
```

```
  feature_1
* master
```

```
33:$ cat hello_world.py
```

```
print('Hello world')
```

```
34:$ git merge feature_1
```

```
Updating ace9ada..d899b2b
```

```
Fast-forward
```

```
  hello_world.py | 1 +
  1 file changed, 1 insertion(+)
```

```
35:$ cat hello_world.py
```

```
print('Hello world')
print('First change')
```

- We are done with the branch called feature_1 and want to integrate it with the deployable version.
- We go back to the master first,
- And then merge the branch with the master
- And then delete the branch called feature_1

```
36:$ git branch -d feature_1
```

```
Deleted branch feature_1 (was d899b2b).
```

```
37:$ git branch
```

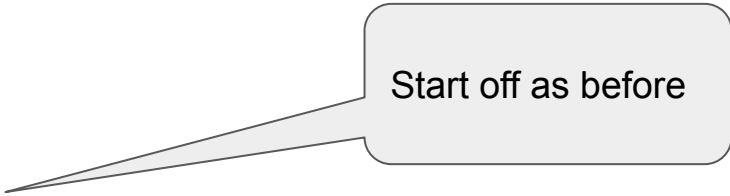
```
* master
$
```

Remote Command line `git`

Scenario

1. Create a local repository
2. Add a Python file to it
3. Make it be looked after by `git`
4. Add everything to your **remote** repository on GitHub
5. Work with the setup
 - a. Create a new branch and work with it
 - b. Make changes, commit and push
 - c. Pull changes from remote
 - d. Merge a branch and delete it locally and on remote

Commands



Start off as before

```
1:$ mkdir Project1.1
```

```
2:$ cd Project1.1
```

```
3:$ vi hello_world.py
```

```
4:$ git init
```

```
Initialized empty Git repository in  
/Users/bianca.schoenphelan/Documents/gitExamples/Project1.1/.git/
```

```
5:$ cat hello_world.py
```

```
print('Hello world')
```

```
6:$ git add hello_world.py
```

```
7:$ git commit -m "Original File"
```

```
[master (root-commit) a40d52d] Original File  
Committer: Bianca SchoenPhelan <bianca.schoenphelan@soc-mbp13-bsp.lan>  
Your name and email address were configured automatically based  
on your username and hostname. Please check that they are accurate.  
You can suppress this message by setting them explicitly. Run the  
following command and follow the instructions in your editor to edit  
your configuration file:
```

```
git config --global --edit
```

After doing this, you may fix the identity used for this commit with:

```
git commit --amend --reset-author
```


```
1 file changed, 2 insertions(+)  
create mode 100644 hello_world.py
```

We need an online repo

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Owner

 BiancaSP ▾

Repository name *

Project1.1 

Great repository names are short and memorable. Need inspiration? How about **ideal-train**?

Description (optional)



Public

Anyone can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.



Initialize this repository with a README

This will let you immediately clone the repository to your computer.

Add .gitignore: **None** ▾

Add a license: **None** ▾



Create repository

Commands cont'd

- gitHub gives you the URL for your project
- `https://github.com/<user>/<repo>`

```
8:$ git remote add origin https://github.com/BiancaSP/Project1.1.git
```

```
9:$ git push -u origin master
```

Everything currently in the local repo will be brought to the remote location

```
Enumerating objects: 3, done.
```

```
Counting objects: 100% (3/3), done.
```

```
Writing objects: 100% (3/3), 260 bytes | 260.00 KiB/s, done.
```

```
Total 3 (delta 0), reused 0 (delta 0)
```

```
To https://github.com/BiancaSP/Project1.1.git
```

```
* [new branch]      master -> master
```

```
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

```
10:$ git status
```

```
On branch master
```

```
Your branch is up to date with 'origin/master'.
```

```
nothing to commit, working tree clean
```

```
11:$ git log --oneline
```

Observe how the output differs from the local example before

```
a40d52d (HEAD -> master, origin/master) Original File
```

The file is online

BiancaSP / Project1.1

Unwatch 1 Star 0 Fork 0

<> Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights Settings

No description, website, or topics provided.

Edit

Manage topics

1 commit

1 branch

0 releases

0 contributors

Branch: master

New pull request

Create new file

Upload files

Find File

Clone or download

Bianca SchoenPhelan Original File

Latest commit a40d52d 1 minute ago

hello_world.py

Help people interested in this repository

BiancaSP / Project1.1

Unwatch 1

Star 0

Fork 0

<> Code

Issues 0

Pull requests 0

Projects 0

Wiki

Security

Insights

Settings

Branch: master Project1.1 / hello_world.py

Find file Copy path

Bianca SchoenPhelan Original File

a40d52d 1 minute ago

0 contributors

3 lines (1 sloc) 22 Bytes

Raw

Blame

History



```
1 print('Hello world')
2
```

Commands cont'd

```
12:$ git checkout -b feature_1
```

Switched to a new branch 'feature_1'

```
13:$ git push origin feature_1
```

Total 0 (delta 0), reused 0 (delta 0)

remote:

remote: Create a pull request for 'feature_1' on GitHub by visiting:

remote: https://github.com/BiancaSP/Project1.1/pull/new/feature_1

remote:

To <https://github.com/BiancaSP/Project1.1.git>

* [new branch] feature_1 -> feature_1

```
14:$ git status
```

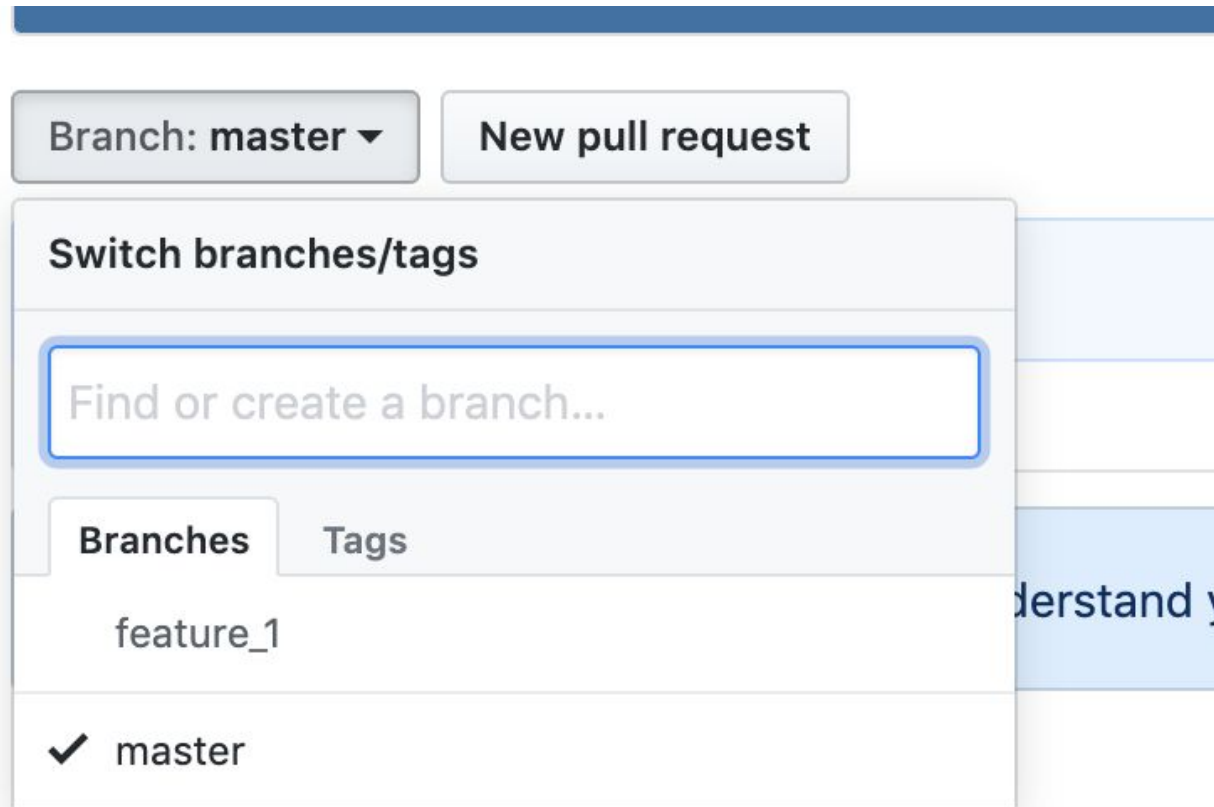
On branch feature_1

nothing to commit, working tree clean

As before

Tell the remote location that we have a new branch.

This is reflected now online



The image shows a GitHub interface. At the top, there is a blue header bar. Below it, there is a button labeled "Branch: master" with a downward arrow, and a button labeled "New pull request". Below these buttons is a dropdown menu titled "Switch branches/tags". Inside the dropdown, there is a search bar with the placeholder text "Find or create a branch...". Below the search bar, there are two tabs: "Branches" and "Tags". Under the "Branches" tab, there is a list of branches: "feature_1" and "master". The "master" branch is selected, indicated by a checkmark to its left.

Branch: master ▼ New pull request

Switch branches/tags

Find or create a branch...

Branches Tags

feature_1

✓ master

Commands cont'd

```
15:$ vi hello_world.py
```

```
16:$ git status
```

On branch feature_1

Changes not staged for commit:

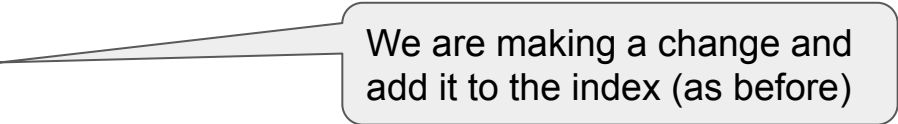
(use "git add <file>..." to update what will be committed)

(use "git checkout -- <file>..." to discard changes in working directory)

```
modified:    hello_world.py
```

no changes added to commit (use "git add" and/or "git commit -a")

```
17:$ git add hello_world.py
```



We are making a change and add it to the index (as before)

Commands cont'd

```
18:$ git commit -m "My first remote change"
```

```
[feature_1 fe595fa] My first remote change
```

```
Committer: Bianca SchoenPhelan <bianca.schoenphelan@soc-mbp13-bsp.lan>
```

Your name and email address were configured automatically based on your username and hostname. Please check that they are accurate. You can suppress this message by setting them explicitly. Run the following command and follow the instructions in your editor to edit your configuration file:

```
git config --global --edit
```

After doing this, you may fix the identity used for this commit with:

```
git commit --amend --reset-author
```

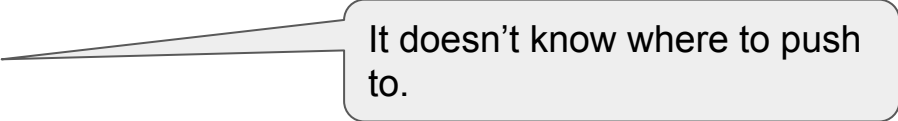
```
1 file changed, 1 insertion(+)
```

```
19:$ git push
```

```
fatal: The current branch feature_1 has no upstream branch.
```

To push the current branch and set the remote as upstream, use

```
git push --set-upstream origin feature_1
```



It doesn't know where to push to.

Commands cont'd

```
20:$ git push --set-upstream origin feature_1
```

```
Enumerating objects: 5, done.
```

```
Counting objects: 100% (5/5), done.
```

```
Delta compression using up to 4 threads
```

```
Compressing objects: 100% (2/2), done.
```

```
Writing objects: 100% (3/3), 323 bytes | 323.00 KiB/s, done.
```

```
Total 3 (delta 0), reused 0 (delta 0)
```

```
To https://github.com/BiancaSP/Project1.1.git
```

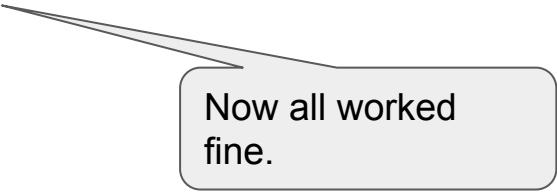
```
    a40d52d..fe595fa  feature_1 -> feature_1
```

```
Branch 'feature_1' set up to track remote branch 'feature_1' from 'origin'.
```

```
21:$ cat hello_world.py
```

```
print('Hello world')
```

```
print('My first remote change')
```



Now all worked fine.

Commands cont'd

```
22:$ git status
```

```
On branch feature_1
Your branch is up to date with 'origin/feature_1'.
```

- It doesn't know we made a change online.
- `git remote update` command can help!

```
nothing to commit, working tree clean
```

```
23:$ git pull
```

```
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/BiancaSP/Project1.1
   fe595fa..8f070e3  feature_1  -> origin/feature_1
Updating fe595fa..8f070e3
Fast-forward
```

Get the latest version online.

```
 hello_world.py | 1 +
 1 file changed, 1 insertion(+)
```

```
24:$ cat hello_world.py
```

```
print('Hello world')
print('My first remote change')
print('hello this is from the github editor')
```

Now all is in synch.

Making a change online

The screenshot shows the GitHub interface for the repository **BiancaSP / Project1.1**. The **Code** tab is selected, displaying the file **Project1.1 / hello_world.py** on the **feature_1** branch. The commit message is **Bianca SchoenPhelan My first remote change**, with **0 contributors**. The file statistics show **4 lines (2 sloc) | 54 Bytes**. The code content is as follows:

```
1 print('Hello world')
2 print('My first remote change')
3
```

A speech bubble points to the code area with the text: **We can edit files online in GitHub.**

Commands cont'd

```
25:$ git checkout master
```

Switched to branch 'master'

Your branch is up to date with 'origin/master'.

```
26:$ git branch
```

```
feature_1
```

```
* master
```

```
27:$ cat hello_world.py
```

```
print('Hello world')
```

```
28:$ git merge feature_1
```

Updating a40d52d..8f070e3

Fast-forward

```
hello_world.py | 2 ++
```


```
1 file changed, 2 insertions(+)
```

```
29:$ cat hello_world.py
```

```
print('Hello world')
```

```
print('My first remote change')
```

```
print('hello this is from the github editor')
```



We are done with feature_1. As before we merge.

Commands cont'd

```
30:$ git branch -d feature_1
```

Deleted branch feature_1 (was 8f070e3).

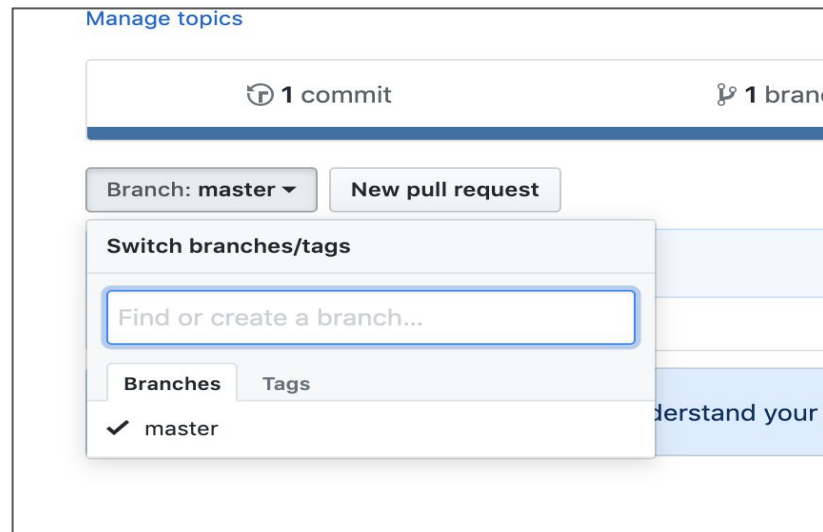
```
31:$ git branch
```

* master

```
32:$ git push origin --delete feature_1
```

To https://github.com/BiancaSP/Project1.1.git
- [deleted] feature_1

We need to delete it online too.



Summary

- ★ Working with git command line
- ★ There is another way to do it all straight out of Pycharm, which we will do next time



Resources

1. Git- a simple Guide, R. Dudler, <https://rogerdudler.github.io/git-guide/>, accessed Sep 2019.
2. Basic git commandes, Freecode, <https://www.freecodecamp.org/news/understanding-git-basics-commands-tips-tricks/>, accessed Sep 2019.
3. Undo Committs, Atlassian, <https://www.atlassian.com/git/tutorials/undoing-changes>, accessed Sep 2019.