



Git and GitHub

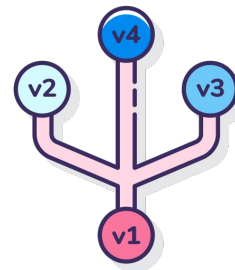
CMPUT 301
Fall 2022



Akalanka Galappaththi

What is Git

- Git is a version control tool.
- Keep track of changes (files).
- Revert back to previous state.



What is GitHub

- Distributed version control system.
- Offers source code management (Git).
- Many other (own) services.



Installing Git

- Windows users: download and run the .exe file
- Mac users: (Homebrew package manager)
- Linux users

<https://git-scm.com/downloads>

Installation guide

<https://www.stanleyulili.com/git/how-to-install-git-bash-on-windows/>

!!! Make sure to select your preferred editor

Installing Git

- Windows users: download and run the .exe file
- Mac users: (Homebrew package manager)
- Linux users

```
$ brew install git  
$ git --version
```

Installing Git

- Windows users: download and run the .exe file
- Mac users: (Homebrew package manager)
- Linux users

```
$ sudo apt update
$ sudo apt install git
$ git --version
```

```
$ sudo yum install git
$ git --version
```


Configure Git

Set up your name and email

```
$ git config --global user.name "John Smith"  
$ git config --global user.email js@gmail.com
```

Set up preferred editor

```
$ git config --global core.editor emacs  
$ git config --global core.editor vim  
$ git config --global core.editor nano  
$ git config --global core.editor "code --wait"
```

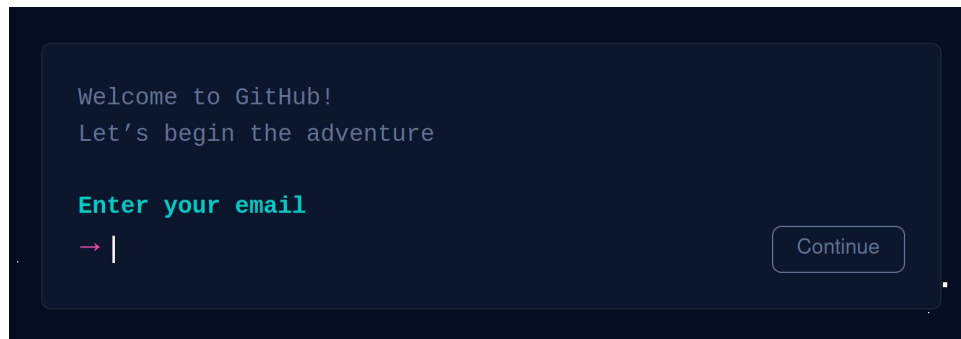
A red starburst callout with multiple points, containing text.

You need an editor
installed in your
computer

Hosting your repo in GitHub

1. First things first: you need a GitHub account (don't have one, go ahead and create one)

https://github.com/signup?ref_cta=Sign+up&ref_loc=header+logged+out&ref_page=%2F&source=header-home



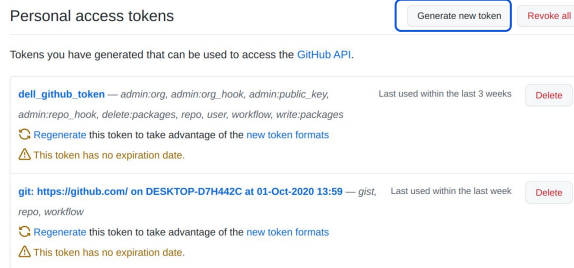
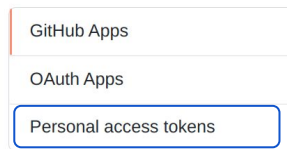
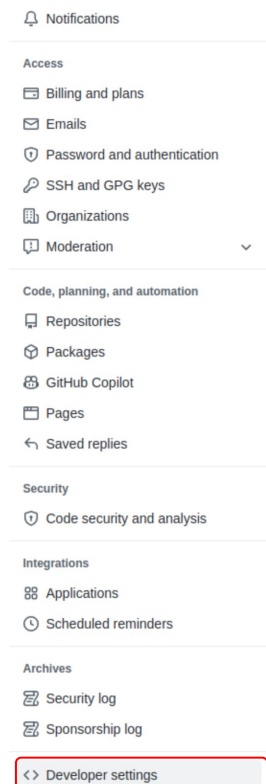
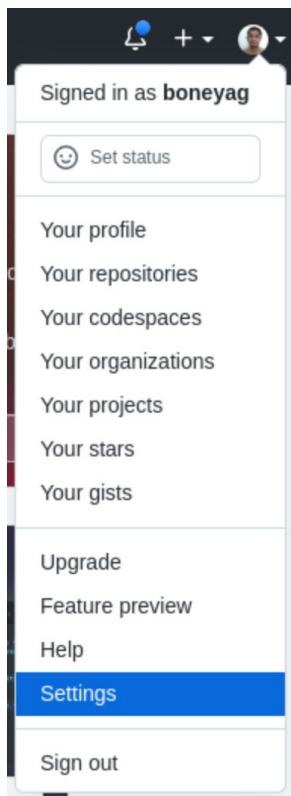
Hosting your repo in GitHub

1. You need personal access tokens to access GitHub (because we are going to use the CMD tool)

Creating a personal access token

You should create a personal access token to use in place of a password with the command line or with the API.

We are going to follow the GitHub docs



I have some tokens created already. In your case it might be empty.

Continue...

New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

CMPUT301

What's this token for?

Expiration *

Custom...

31/12/2022

Select scopes

Scopes define the ac

December 2022						
M	T	W	T	F	S	S
28	29	30	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1
2	3	4	5	6	7	8

- repo
 - repo:status
 - repo_deployments
 - public_repo
 - repo:invite
 - security_events
- workflow
- write:packages
 - read:packages
- delete:packages
- admin:org
 - write:org
 - read:org
 - manage_runners:org

Not super important, but use a meaningful name so that you know the purpose

Important: keep until you finish the course

You may read the scope for more information. But these three option should enough for the project.

Continue...

<input type="checkbox"/>	admin:gpg_key	Full control of public user GPG keys (Developer Preview)
<input type="checkbox"/>	write:gpg_key	Write public user GPG keys
<input type="checkbox"/>	read:gpg_key	Read public user GPG keys


Generate token

Cancel

Click on the generate token

Tokens you have generated that can be used to access the [GitHub API](#).

Make sure to copy your new personal access token now. You won't be able to see it again!

✓ ghp_IqIMN0ZH6z0wIEB4T9A2g4EHMy8Ji42q4HA5		Enable SSO ▾	Delete
--	---	--------------	--------

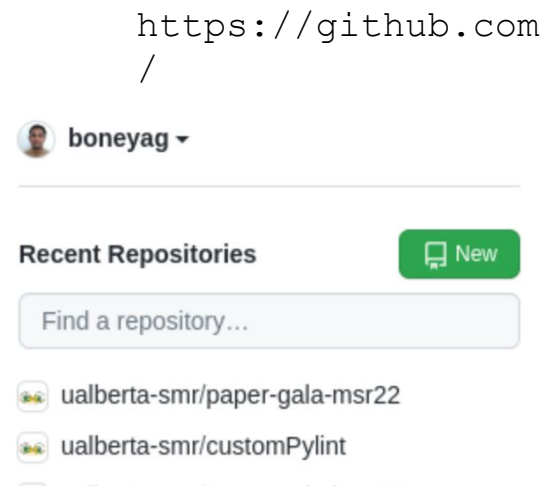
Copy the token to a file,
because you will not be able to
see it!!!!

When using `git push/pull`, you use the PAT instead of password.

If you need to store your PAT, follow this [discussion](#) in SO (at your own risk).

Create a repo in GitHub

- Visit the URL on your browser.
- Create a repo.



Create a repo in GitHub

Repository template

Start your repository with a template repository's contents.

No template ▾

Owner *

 boneyag ▾

Repository name *

test-301-tue ✓

Great repository names are short and memorable. Need inspiration? How about [sturdy-journey?](#)

Description (optional)

Test repo for Tuesday lab

 **Public**

Anyone on the internet can see this repository. You choose who can commit.

 **Private**

You choose who can see and commit to this repository.

Initialize this repository with:

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

Add a README file

This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: None ▾

Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

License: None ▾

 You are creating a public repository in your personal account.

Create repository

Short memorable name

Who could see your repo

Automatically create some files

Use the https opt. (ssh is out of the scope)

Quick setup — if you've done this kind of thing before

or HTTPS SSH



Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

Your username

Your repo name

Copy URL and paste on the terminal
(make sure you to change dir to a
desired location)

...or create a new repository on the command line

```
echo "# test-301-tue" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/boneyag/test-301-tue.git
git push -u origin main
```



...or push an existing repository from the command line

```
git remote add origin https://github.com/boneyag/test-301-tue.git
git branch -M main
git push -u origin main
```



...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code

Clone a Git repo

```
akalanka@akalanka-ThinkPad: ~/GitDemo/test-301Tue
akalanka@akalanka-ThinkPad:~$ mkdir GitDemo
akalanka@akalanka-ThinkPad:~$ cd GitDemo
akalanka@akalanka-ThinkPad:~/GitDemo$ git clone https://github.com/boneyag/test-301Tue.git
Cloning into 'test-301Tue'...
warning: You appear to have cloned an empty repository.
akalanka@akalanka-ThinkPad:~/GitDemo$ ls
test-301Tue
akalanka@akalanka-ThinkPad:~/GitDemo$ cd test-301Tue/
akalanka@akalanka-ThinkPad:~/GitDemo/test-301Tue$ ls
akalanka@akalanka-ThinkPad:~/GitDemo/test-301Tue$
```

1. Make a dir
2. Change the dir
3. Use the command `git clone`
4. Copy git URL after
5. Hit return

****Use `git clone` when copy the repo to your computer for the first time. After that we use `git pull`.**

Add a file to the repo locally

```
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4$ git clone https://github.com/boneyag/test-301-tue.git
Cloning into 'test-301-tue'...
warning: You appear to have cloned an empty repository.
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4$ ls
test-301-tue
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4$ cd test-301-tue/
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue$ ls
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue$ code README.md
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue$ ls
README.md
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue$ more README.md
This is a test repository for CMPUT 301 Tuesday lab (Fall 2022)
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue$ git status
On branch main

No commits yet

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

nothing added to commit but untracked files present (use "git add" to track)
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue$
```

```
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue$ git add README.md
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue$ git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   README.md

akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue$ git commit -m "Add a readme
file"
[main (root-commit) 077d32b] Add a readme file
1 file changed, 1 insertion(+)
create mode 100644 README.md
```

```
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue$ git push origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 275 bytes | 275.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/boneyag/test-301-tue.git
 * [new branch]    main -> main
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue$
```

Create a repo locally

```
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4$ mkdir test-301-tue-2
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4$ ls
test-301-tue  test-301-tue-2
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4$ cd test-301-tue-2
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue-2$ code README.md
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue-2$ more README.md
Creating a repo locally
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue-2$ git status
fatal: not a git repository (or any of the parent directories): .git
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue-2$
```

```
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue-2$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/akalanka/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue-2/.git
```

```
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue-2$ git branch -m main
```

```
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue-2$ git status
On branch main

No commits yet

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

nothing added to commit but untracked files present (use "git add" to track)
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue-2$
```

```
code README.md
git init
git status
git add README.md
git status
git commit -m "<message>"
git remote add origin <URL>
git push origin master
```

...or push an existing repository from the command line

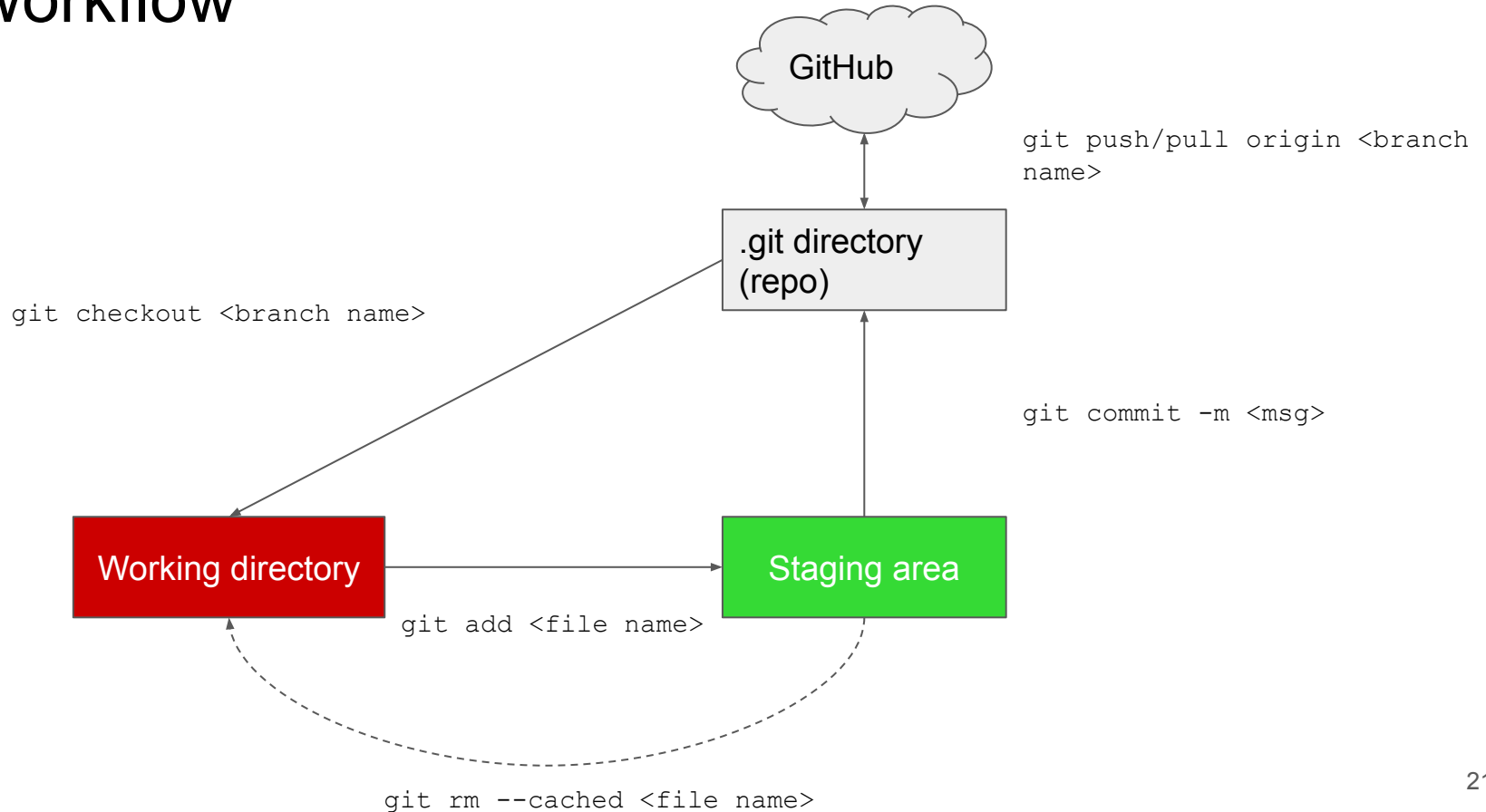
```
git remote add origin https://github.com/boneyag/test-301-tue-2.git
git branch -M main
git push -u origin main
```



```
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue-2$ git remote add origin https://github.com/boneyag/test-301-tue-2.git
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue-2$ git remote -v
origin https://github.com/boneyag/test-301-tue-2.git (fetch)
origin https://github.com/boneyag/test-301-tue-2.git (push)
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue-2$
```

```
akalanka@Akalanka-ThinkPad:~/Documents/TA/CMPUT301_Fall22/Lab4/test-301-tue-2$ git push origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 238 bytes | 238.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/boneyag/test-301-tue-2.git
 * [new branch]      main -> main
```

Git workflow




Practice the common commands

- Create two new text files and push those to Github repo
 - `echo "Test 1" >> test1.txt`
 - `echo "Test 2" >> test2.txt`
 - `git add - A`
 - `git commit -m "<message>"`
 - `git push origin master`
- Useful git commands
 - `git pull origin <branch>`
 - `git reset - remove files from the staging area`
 - `git rm --cached <filename> - remove a file from working index`
 - `git rm -f <filename> - remove a file forcefully (-f)`
 - `git rm -rf <dir name> - remove a directory forcefully (-r recursively)`
 - `git log - view commit history`
 - `git clone <https URL>`

Merge conflicts

Ex: Modifying files in the project.

- TM1: change file1
- TM2: change file2
- Both push changes 
- TM1: change file1 -> push
- TM2: change file1, file2 -> try to push



Let's get your hands dirty -- demo a merge conflict

Leave the current repo dir (cd ..), open two terminals (pretend those as two users of the repo).
Make sure to follow the order of execution of commands.

Terminal 1

```
1  mkdir t1
2  cd t1
3  git clone <your repo URL>
4
5
6
7
```

Terminal 2

```
mkdir t2
cd t2
git clone <your repo URL>
cd test-301-tue
mkdir one
echo "Lin1" >> one/file1.txt
```


Let's get your hands dirty -- demo a merge conflict

```
8           git status
9
9           git add one/file1.txt
10
10          git commit -m "informative-message"
11
11          git push origin main
12
12 cd test-301Tue
13
13 git pull origin main
14
14 mkdir two
15
15 code two/file2.txt
16
16 git add two/
```

Let's get your hands dirty -- demo a merge conflict

```
17 git commit -m "<msg>"
```

```
18 git push origin main
```

No conflict as
users pushed
different files

```
19 cd one
```

```
20 code file1.txt # add two lines
```

```
21 git add file1.txt
```

```
22 git commit -m "<msg>"
```

```
23 git push origin main
```

Will not push
until you pull
remote
changes

```
24 git fetch origin
```

```
25 git merge origin main
```

Let's get your hands dirty -- demo a merge conflict

26

```
git push origin main
```

27

```
git pull origin main
```

Need to pull changes from origin

28

```
code two/file2.txt
```

Add two more lines

No conflict as
users pushed
different files

29

```
git add two/file2.txt
```

30

```
git commit -m "<msg>"
```

31

```
git push origin main
```

32

```
cd two
```

33

```
code file2.txt
```

Add two more lines

34

```
git add two/file2.txt
```

Let's get your hands dirty -- demo a merge conflict

```
35         git commit -m "<msg>"
36
37         git push origin main
38
39         git fetch origin
40
41         git merge origin main
42         # Fix the conflict as you see fit #
43         git add two/
44         git commit -m "<msg>"
45         git push origin master
```

Need to pull changes from origin

Conflict

Git branches

scribble 2 branches 0 tags

Go to file Add file Code

Switch branches/tags

Find or create a branch...

Branches Tags

main

✓ scribble default

View all branches

f980b39 26 days ago 19 commits

Added CodeSearchNet summary	4 months ago
How to start the code search project	4 months ago
Update README.md	26 days ago

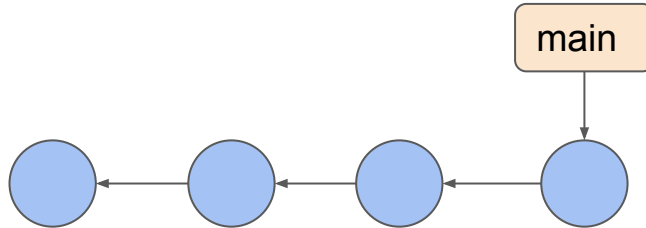
code-search-lit-review

Links to source code repositories

Git branches

What is a branch?

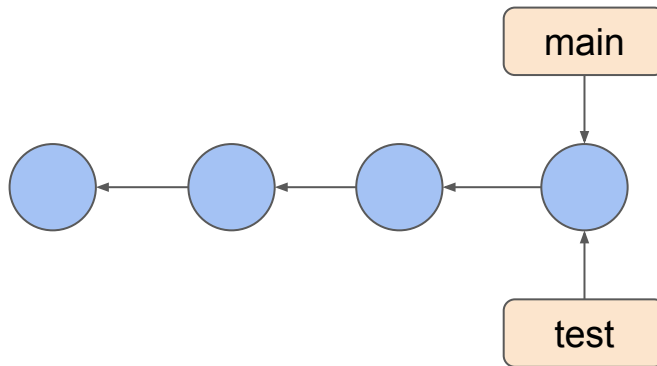
- Branch shows the evolution of your project (commits)
- Each commit has SHA-1 value which allows you to revert back to that state.



Git branches

- Default branch -- main
- Nothing special about this branch.
 - You can rename it.
 - Nobody bother to do that, so that it remains with the default name.
- Create a branch in command line

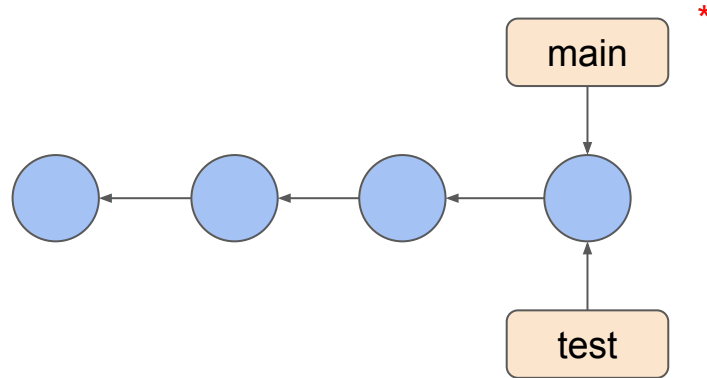
```
git branch test
```



Git branches

- Current working branch
- Change the branch

```
git status
```

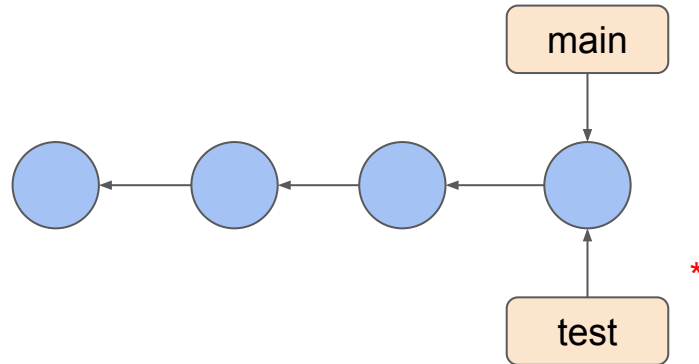


Git branches

- Current working branch
- Change the branch

```
git status
```

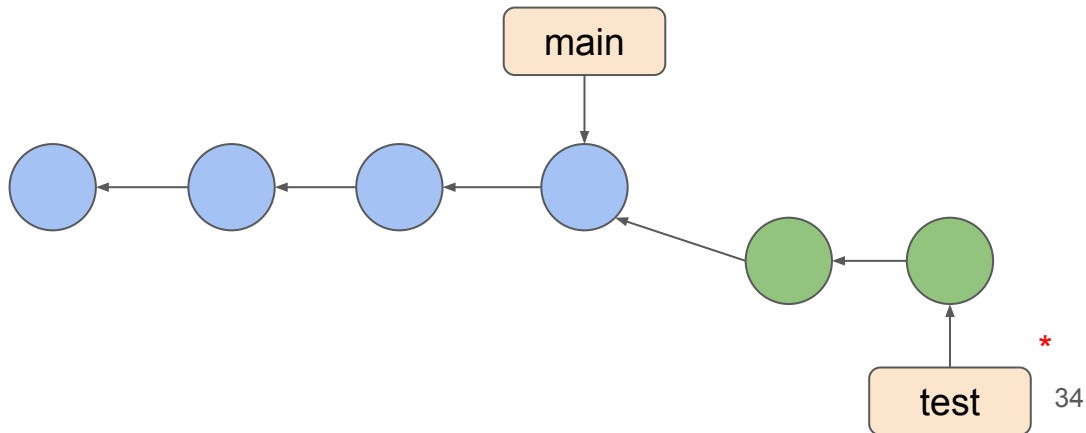
```
git checkout test
```



Git branches

- Do some work on test branch

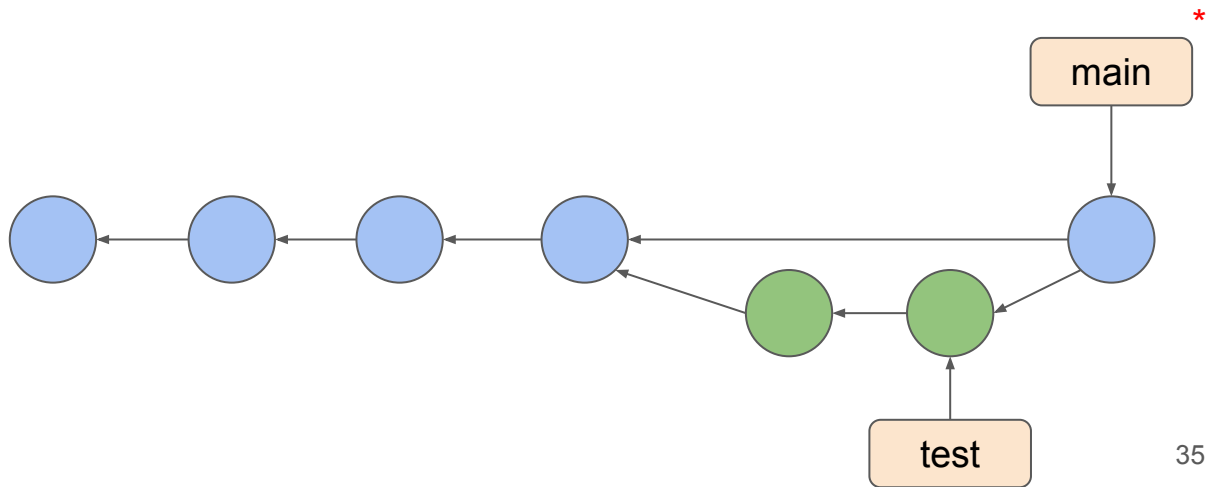
```
git checkout test
code two/file2.txt
git add two/
git commit -m "<msg>"
git push origin test
code one/file2.txt
git add one/
git commit -m "<msg>"
git push origin test
```



Git branches

- Merge test to master

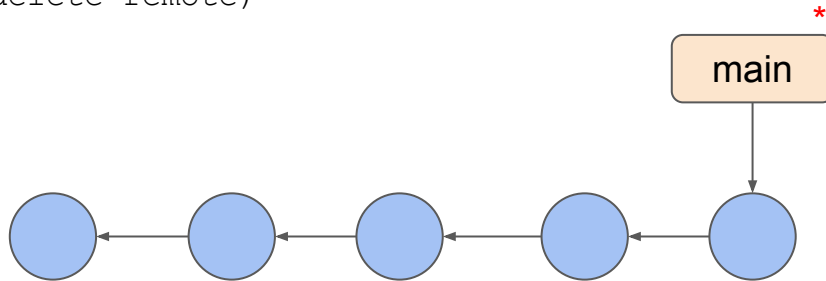
```
git checkout main  
git merge test
```



Git branches

- Merge test to master

```
git checkout master
git merge test
git branch -d test (delete local)
git push -d origin test (delete remote)
```



Skip files from adding to the staging area

.gitignore is a special file that contains file patterns that skip when adding files to the staging area

<https://github.com/github/gitignore>

```
# IntelliJ
*.iml
.idea/workspace.xml
.idea/tasks.xml
.idea/gradle.xml
.idea/assetWizardSettings.xml
.idea/dictionaries
.idea/libraries
```

GitHub organization

The diagram illustrates the steps to create a GitHub organization:

- Step 1:** The user's profile menu is open, showing options like "Signed in as boneyag", "Set status", "Your profile", "Your repositories", "Your codespaces", "Your organizations", "Your projects", "Your stars", "Your gists", "Upgrade", "Feature preview", "Help", "Settings" (highlighted), and "Sign out".
- Step 2:** The "Account settings" page is shown, with "Organizations" selected in the left sidebar. Other settings include Profile, Account, Appearance, Account security, Billing & plans, Security log, Security & analysis, Sponsorship log, Emails, Notifications, Scheduled reminders, SSH and GPG keys, Repositories, Packages, Saved replies, and Applications.
- Step 3:** The "Organizations" page is shown, displaying existing organizations and a "New organization" button. The organizations listed are:
 - SibyLab** member and collaborator on 2 repositories (Leave)
 - ualberta-smr** member and collaborator on 5 repositories (Leave)

Below the organizations list, there is a section titled "Transform account" with the message: "You cannot transform this account into an organization until you leave all organizations that you're a member of." and a button "Turn boneyag into an organization".

Continue...

Tell us about your organization

Set up your organization

Organization account name *

This will be the name of your account on GitHub.
Your URL will be: <https://github.com/>

Contact email *

This organization belongs to: *

My personal account

I.e., boneyag (Akalanka)


A business or institution

For example: GitHub, Inc., Example Institute, American Red Cross

Verify your account

Please solve this puzzle so we know you are a real person

Verify



Next

By creating an account, you agree to the [Terms of Service](#). For more information about GitHub's privacy practices, see the [GitHub Privacy Statement](#). We'll occasionally send you account-related emails.

- Choose free option
- Provide required information
- Finish the process
- Add team members as collaborators
- Do the class participation exercise by creating a new repo
- For the project, create a separate repo