

Git

R11922109 趙雋同

Preparation

Environment

- Linux or MAC OS is strongly recommended.
- Windows
 - use git bash / bash / powershell.
 - Install linux (ubuntu is suggested) on virtual machine or dual system.
 - connect to CSIE workstation.
- connect to CSIE workstation
 - ssh [your school id]@linux[machine number].csie.(org/ntu.edu.tw)
 - e.g. ssh **r11922109@linux10.csie.ntu.edu.tw**

Useful Terminal Command (Linux/MacOS)

command	description	example
cd	Changes the directory of the command line path.	cd "path/to/directory/"
ls	Lists the contents of a directory.	ls "path/to/directory/"
cp	copy file	cp "filename" "newfilename"
mv	move a file	mv "filename" "path/to/new/file/location"
mv	rename a file	mv "path/to/filename" "path/to/newfilename"
rm	remove a file	rm "path/to/filename"
mkdir	create a directory	mkdir "path/to/new/directory"

Install Git on Linux (ubuntu)

- `sudo apt update`
- `sudo apt install git`

Install vim on Linux (ubuntu)

- `sudo apt update`
- `sudo apt install vim`

Suggest Editor -- Vim

- Common commands
 - “i”: insert (enter editing mode)
 - “o”: insert on the new line (enter editing mode)
 - “\esc”: escape from editing mode
 - “:w”: write to file
 - “:q”: quit vim
 - “:wq”: write and quit
 - “[number] yy”: copy [number] lines
 - “p”: paste copied lines on the current line
 - “[number] dd”: delete [number] lines
 - “v + [→, ←]”: underline messages
 - “u”: undo (need to set up undo buffer in vimrc)
 - “\ctrl r”: redo

- Make vim more beautiful

- The image shows a Vim editor window with a file named 'colors.vim'. The file contains a series of 'let' statements defining syntax highlighting rules. The first rule is 'let colors_name = "lucius"', which sets the theme to 'lucius'. The second rule is 'let s:style = &background', which sets the background color to the current background color. The third rule is 'let s:high_contrast = g:lucius_high_contrast', which sets the high contrast flag to the value of 'g:lucius_high_contrast'. The fourth rule is 'let s:high_contrast_bg = g:lucius_high_contrast_bg', which sets the high contrast background color to the value of 'g:lucius_high_contrast_bg'. The fifth rule is 'let s:high_contrast_bg = 0', which sets the high contrast background color to 0. The sixth rule is 'let s:high_contrast_bg = 0', which sets the high contrast background color to 0. The seventh rule is 'let s:use_bold = g:lucius_use_bold', which sets the use bold flag to the value of 'g:lucius_use_bold'. The eighth rule is 'let s:use_bold = 1', which sets the use bold flag to 1. The ninth rule is 'let s:use_underline = g:lucius_use_underline', which sets the use underline flag to the value of 'g:lucius_use_underline'. The tenth rule is 'let s:use_underline = 1', which sets the use underline flag to 1. The editor shows the 'lucius' theme is active, with various syntax elements highlighted in different colors. The status bar at the bottom shows '11% <p>[-][RO][help][dos]233,1' and '4% Highlight test [unix] 1,1'.

<https://camo.githubusercontent.com/11a2b8c9e2544000911b0f795e9c4e3175a9c757/687474703a2f2f692e696d6775722e636f6d2f65373069392e706e67>

CSIE resources

Printer

- Printing quota:
 - Each CSIE student has 500 dollars for printing every semester (can't be cumulated).
 - NTU also gives each student 100 dollars for printing every semester (can be cumulated).
- Therefore, you can print your HW without money.

CSIE printer introduction

- Log in <https://printing.csie.ntu.edu.tw/user> .

Use your csie workstation account and password. You should received an email containing your workstation account and password after the enrollment.

PaperCut NG

 摘要

 費率

 轉帳

 交易歷史記錄

 近期的列印工作

 工作待決釋放


 網頁列印

網頁列印

Web Print is a service to enable printing for laptop, wireless and unauthenticated users without the need to install print drivers. To upload a document for printing, click Submit a Job below.

提交工作。


提交時間	印表機	文件名稱	頁數	費用	狀態
沒有活躍的工作					

 摘要 費率 轉帳 交易歷史記錄 近期的列印工作 工作待決釋放 網頁列印

網頁列印

1. 印表機   
2. 選項 3. 上傳

選擇一台印表機：

印表機名稱 ↑	位置/部門
<input type="radio"/> printing2\Ariel	next to 209 (two-sided)
<input type="radio"/> printing2\Elsa2	next to R209 Ariel
<input type="radio"/> printing2\Elsa2(One_Sided)	next to R209(One_Sided)

[« 返回活躍的工作](#)[2. 列印選項及帳戶選擇 »](#)

The printer is next to 209.

Workstation

- csie workstation:

ssh [your school id]@linux[1-10].csie.ntu.edu.tw

```
ubuntu ~ ssh r11922109@linux10.csie.ntu.edu.tw
r11922109@linux10.csie.ntu.edu.tw's password:
#####
#      Public Domain Workstation Lab (R217).      #
#####
#  UNIX Login Service:                          #
#    FreeBSD - bsd1                             #
#    Linux   - linux1, linux2, linux3, ... linux15 #
#              - [NEW!] meow1, meow2 (with GPU)    #
#              - oasis1, oasis2, oasis3 (non-computing) #
#    Run `ws-status` for current machine status  #
#                                                  #
#  Office open time:                             #
#    08:30 ~ 17:00, otherwise please use accesscards #
#                                                  #
#  Contact information:                          #
#    Web:   https://wslab.csie.ntu.edu.tw/        #
#    E-Mail (linux): ta217@csie.ntu.edu.tw        #
#    E-Mail (bsd)  : lantw44@csie.ntu.edu.tw      #
#                                                  #
##### Last Update: Sep 26 2018 ###
Last login: Fri Oct 28 12:01:49 2022 from 140.112.29.128
mail: /var/spool/mail/r11922109: No such entry, file or directory
西元 2022 年 10 月 28 日 (週五) 12 時 02 分 04 秒 CST
r11922109@linux10 [-] █
```

- All tips and information are in <https://wslab.csie.ntu.edu.tw/>

Git Intro

Problem Definition

- Before we complete our codes or maybe homeworks, reports etc, we might
 - modify the contents several times
 - regret and press `ctl + z`
 - accidentally make situation worse



EVERY DESIGNER IN THIS WORLD



New.**psd**



Newfinal.**psd**



Newfinalfinal.**psd**



Newfinalest
final.**psd**



Newfinalest
final forsure.**psd**



Newfinalestf ckthis
shitfinal.**psd**

Benefits of Version Control

- A complete long-term change history of every file
 - records creation and deletion of files as well as edits to their contents
 - includes the author, date and written notes on the purpose of each change
 - able to go back any previous version you desired
- Branching and merging
 - keeps multiple streams of work independent from each other
 - provides the facility to merge works back together
- Traceability
 - being able to annotate each change with a message describing the purpose and intent of the change can help not only with root cause analysis



- A distributed version control system
- Free and open source
- Created by Linus Torvalds

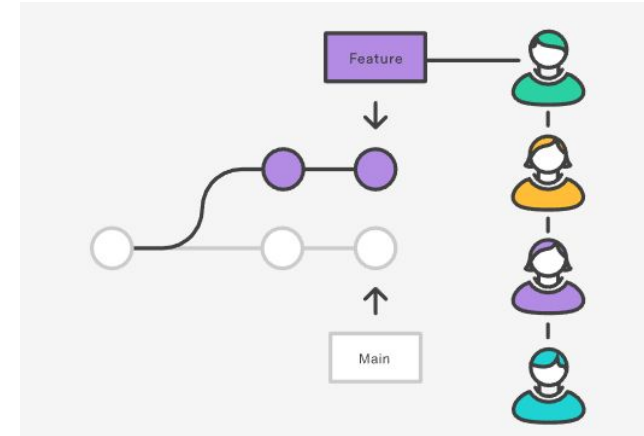
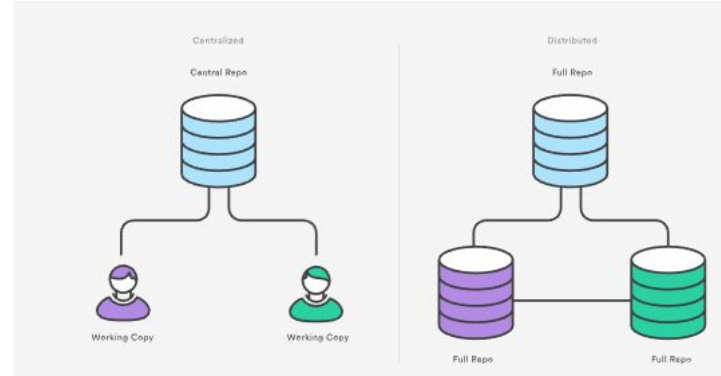
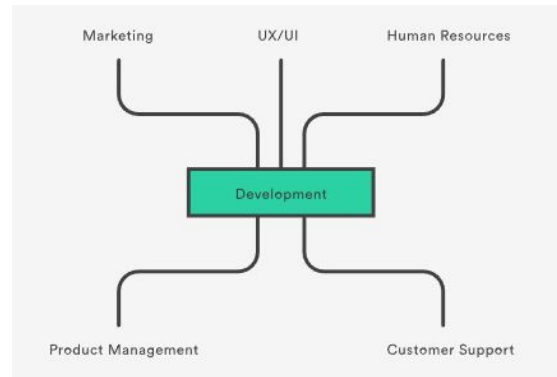
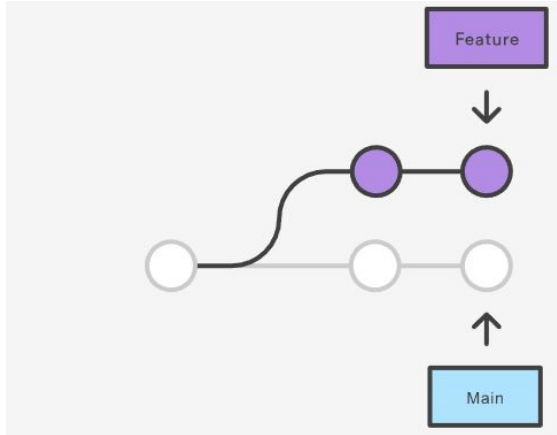


What does Git do

- A distributed version control system
- Free and open source
- Created by Linus Torvalds



More About Git



Github intro



- People use GitHub to build the most advanced technologies in the world
 - It makes tools that use Git
- There's a whole set of tools on GitHub that can help you
- **You put codes on GitHub and show them to the world**
- If you are to become a software engineer,
then GitHub is one of the basic tools you **MUST learn**.



- Sign up for your account with your school email (or your email)
- If you have time, apply education pack for more functions, e.g.
 - Create private repository
 - Use advanced insight tools
- Note that this account will put all your source codes in the future, please be careful about everything you do on it.
 - For example, the username should be a decent one.
- Due to new policy, password is not allowed anymore
 - Personal access token:
<https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token>
 - SSH key: <https://ithelp.ithome.com.tw/articles/10205988>

Tutorial -- Create Repository

- Each of your project is put in a repository

Create a new repository

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

Owner *

Repository name *

 ander1119

/

Great repository names are short and memorable. Need inspiration? How about [supreme-octo-robot?](#)

Description (optional)



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

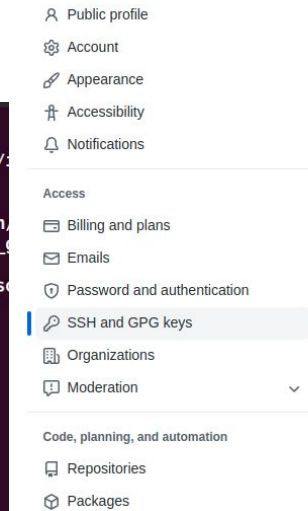
 You are creating a public repository in your personal account.

Create repository

Tutorial -- Add SSH key

- Create SSH key in local
- Copy public key and paste it onto GitHub SSH key management

```
ubuntu ~ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/anderson/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/anderson/.ssh/id_rsa.
Your public key has been saved in /home/anderson/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:oYES29Gpc3TI5AmcWSfD0LSiyleWr5G6FuEUB6ales8 anderson@cmLab
The key's randomart image is:
+----[RSA 3072]----+
|.o*%*..|
|+ B*oB*|
|o = o*|=|
|o .+=+oo|
|.oo.+S|
|o +oo o|
|o E.o |
|... o|
|.o..|
+----[SHA256]-----+
ubuntu ~ cat /home/anderson/.ssh/id_git_test.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCxcTE8WqKeYtpqJxdafsrUuVryVDeRJYF1BC5mYtSWyEn30zxa3sTshJ2udv0ZtpwFAq+BRFKMD5KuW1SgWVQ8RNxbpPLZfvYtjjQ1b8BtwyZVwqrPE2jffTRb49AK1x
eGhP3J0RdzilrqW40zTXPiolpdaNrLBJFMeMIDcfLaQwI0m0I2lmWmU66mZtGfUut1a5/ZgwdpIGaooxpi9zT7ZbUSG6VypkL0wz5B3F4jqi7fUH6M9JampQ0JCnq3LUUEqBr8YLuyLhVE73Z4Dla50KYxjiPRRrwoXW3Q
T7LTprGmWBAD85trfH9waQ/IuqZB5gVmVsJvrVXyeQZCEkss5nU2V5GWFptnJu6fZsHtS5Di2XQmeUOKFJvoddK0YU7nf575d66vyz5sRRZnd+oA00st4N4QI8e00LdZWwLmoYpNBkj5R41Grw/FpIp2BwodRLnqD+HUL
tJqRtcBtpHXFRb/12Lud+1wQnPrGiWtaw6TYKbcVZG0jK/0nAsqWc= anderson@cmLab
```



SSH keys / Add new

Title

Key type

Authentication Key

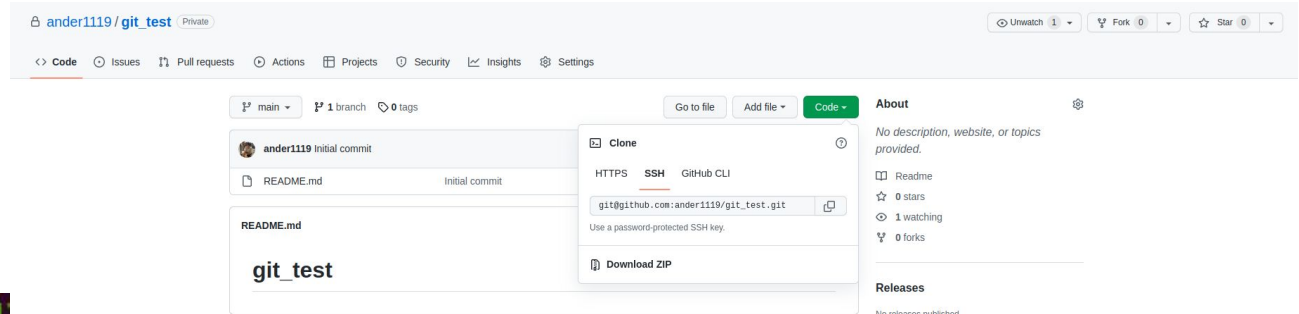
Key

Begins with 'ssh-rsa', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', 'ecdsa-sha2-nistp521', 'ssh-ed25519', 'sk-ecdsa-sha2-nistp256@openssh.com', or 'sk-ssh-ed25519@openssh.com'

Add SSH key

Tutorial -- Clone Repository to Local

- Copy the repo url
- Enter “git clone [url]” command in your local terminal



```
ubuntu ~/Downloads g
Cloning into 'git_test'
remote: Enumerating objects
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```


Add my teammate

11 requests


▶ Actions

📁 Projects

📖 Wiki

🛡 Security

📊 Insights

 Settings

Options

Manage access

Security & analysis

Branches

Webhooks

Notifications

Integrations

Deploy keys

Autolink references


Actions

Secrets

Moderation settings


Who has access

PUBLIC REPOSITORY




This repository is public and visible to anyone.
[Manage](#)

DIRECT ACCESS



0 collaborators have access to this repository. Only you can contribute to this repository.

Manage access





You haven't invited any collaborators yet

Invite a collaborator


Invite my teammate


The teammate should received an email.





Add a collaborator to **git_test**

 shelley1214



Shelley1214
Invite collaborator

Select a collaborator above

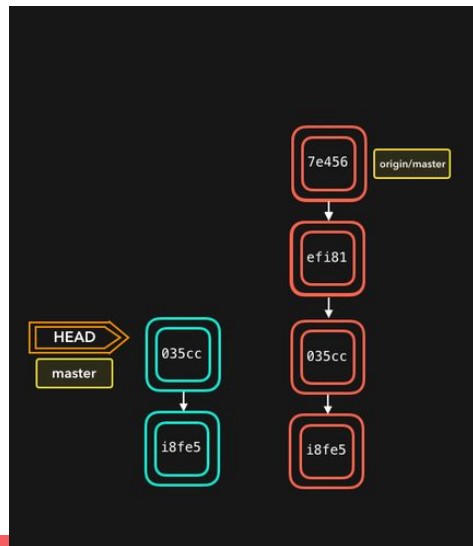


V.S.



Local v.s. Remote

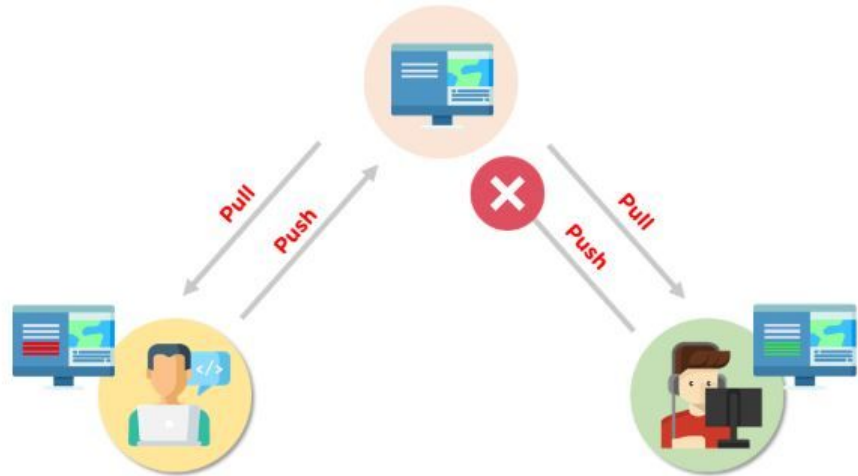
- **Local Repo** is an Git repository that stored on **your own computer**
 - you can modified and commit changes to it
 - nobody can see the changes until you push them to remote
 - the blue parts
- **Remote Repo** is an Git repository that stored on some **remote server (GitHub)**
 - used by teams as a central repository
 - everyone pushes changes from his/her local repo to remote repo
 - everyone pulls changes from remote rep to his/her own local repo
 - the orange parts (usually named as **origin/[branch name]**)



credits: CS visualized from Lydia Hallie

Local and Remote

- Most of version control related work happens in local repo
 - staging, committing, viewing status or showing log
 - it's your own project, you are the only person putting into
- When it comes to sharing data with teammates, remote repo comes into play
 - deals with conflict of contents
 - how ?



Git Command

What does Git do ? (Reviews)

- Manage projects with **Repositories**
- **Clone** a project to work on a local copy
- Control and track changes with **Staging** and **Committing**
- **Branch** and **Merge** to allow for work on different parts and versions of a project
- **Pull** the latest version of the project to a local copy
- **Push** local updates to the main project

Tutorial -- Show status

- First create a new file at local, then type **git status** to show the working tree status

```
ubuntu ~/Downloads/git_test main ?1 git status
On branch main
Your branch is up to date with 'origin/main'.

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

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


Tutorial -- Add modified files

- To add the modification of your file, type **git add <files>...**
- Then you can check working tree status again

```
ubuntu ~/Downloads/git_test main ?1 git add test.cpp
ubuntu ~/Downloads/git_test main +1 git status
```

On branch main

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

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

new file: test.cpp

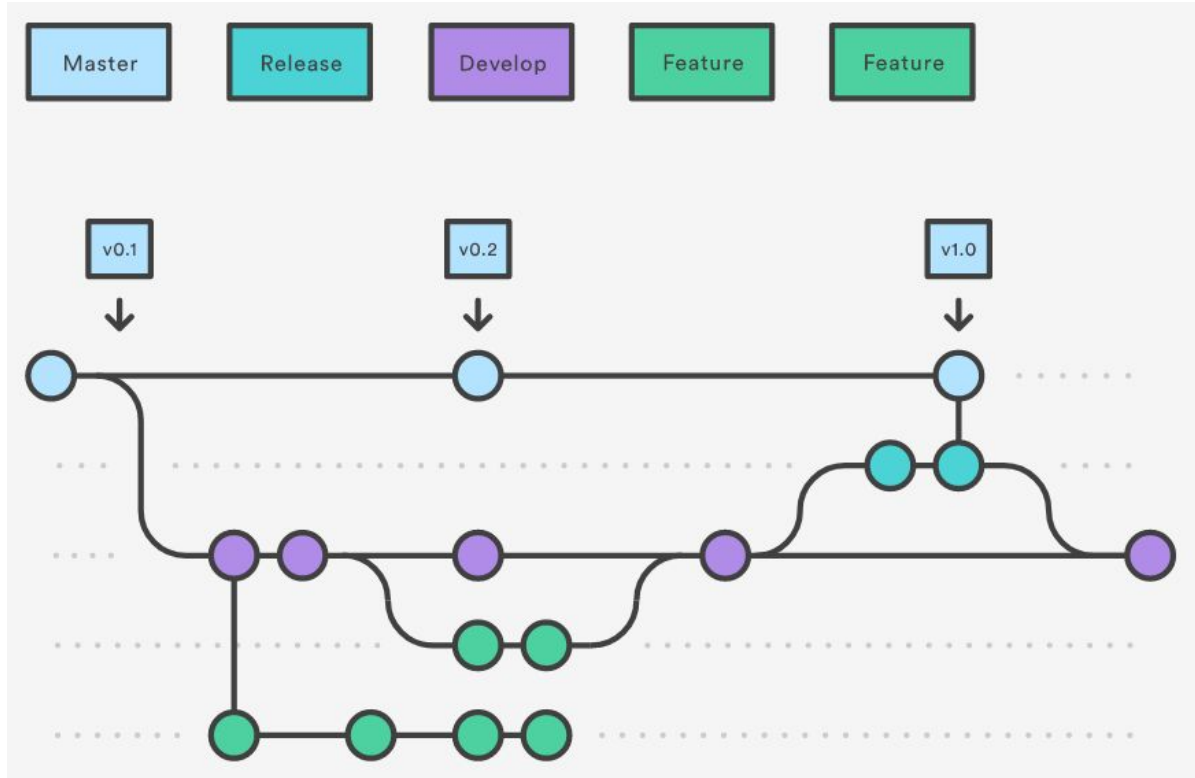
Tutorial -- Commit to your own repo

- To commit it into local repository, type `git commit -m "description message"`

```
ubuntu ~/Downloads/git_test main +1 git commit -m "create test.cpp"
[main 72d6016] create test.cpp
1 file changed, 6 insertions(+)
create mode 100644 test.cpp
ubuntu ~/Downloads/git_test main >1 git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
    (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

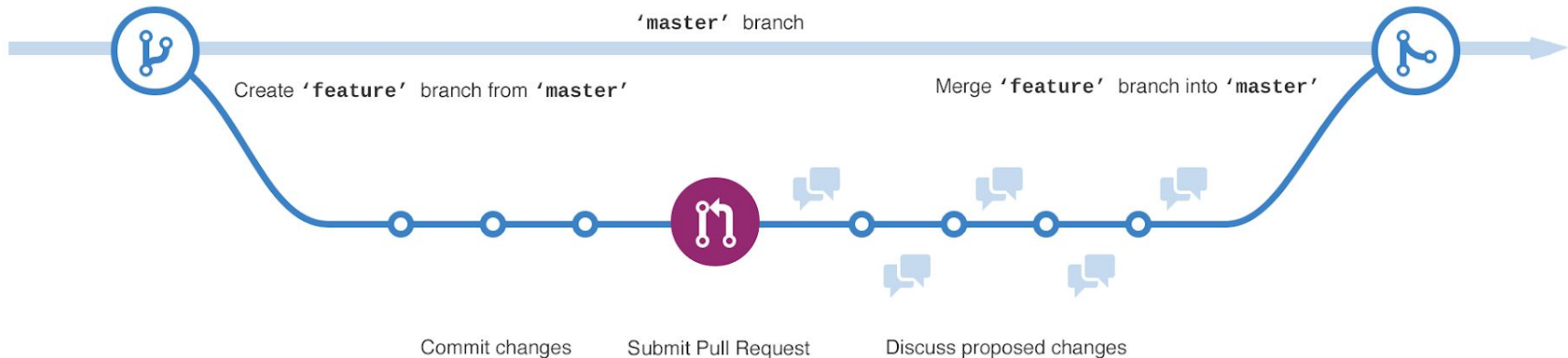
Tutorial -- Branch



- master is the product
- Add features to develop branch
- After finishing and testing the feature, merge to master

Tutorial -- Create a Branch

- Branching is the way to work on **different versions** at one time
- We use branches to experiment and make edits before committing them to master (main).
- master (main) branch should be clean.
- At branch master (main)
 - create branch: **git branch [branch name]**
 - delete branch: **git branch -d [branch name]**
 - change to the branch: **git checkout [branch name]**



Tutorial -- Protect “main” Branch

- Important when developing big projects
 - Enhance code quality without creating artificial obstructions to effective collaboration
- Setting → Branches → add rule

Branch protection rule

Branch name pattern

Applies to 1 branch

main

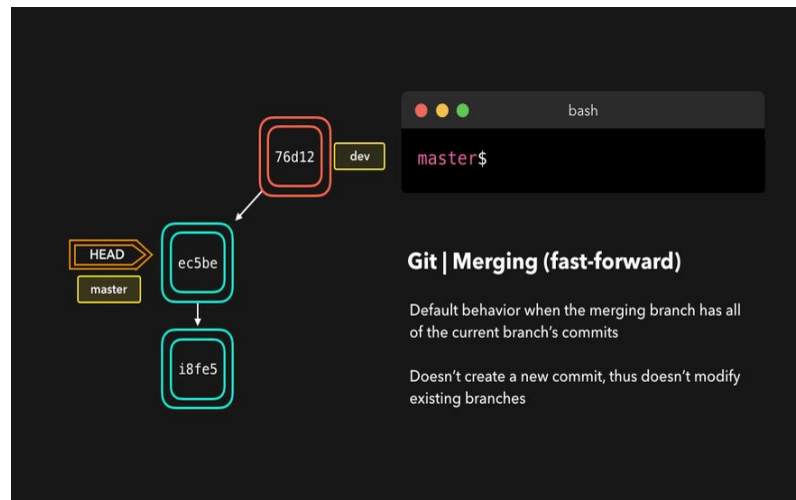
Protect matching branches

☒ **Require pull request reviews before merging**
When enabled, all commits must be made to a non-protected branch and submitted via a pull request with the required number of approving reviews and no changes requested before it can be merged into a branch that matches this rule.

Required approving reviews: 1 ▼

Tutorial -- Merge

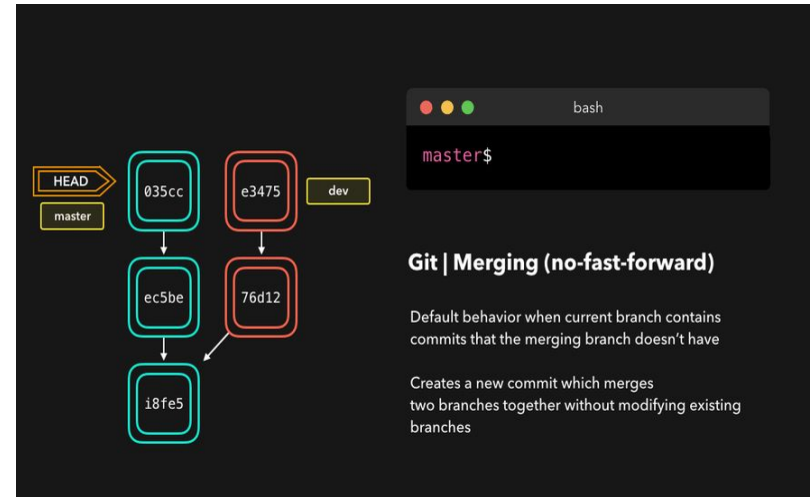
- Put a forked history back together, integrate them into a single branch
- In most use cases, **git merge [target branch]** is used to combine two branches
- Merge could be classified into
 - fast-forward
 - no-fast-forward



credits: [CS visualized from Lydia Hallie](#)

Tutorial -- Merge

- In **no-fast-forward** case, Git will find a common base commits between two branches, then create a new merged commit
- The merged commit contains changes of each queued commit sequence

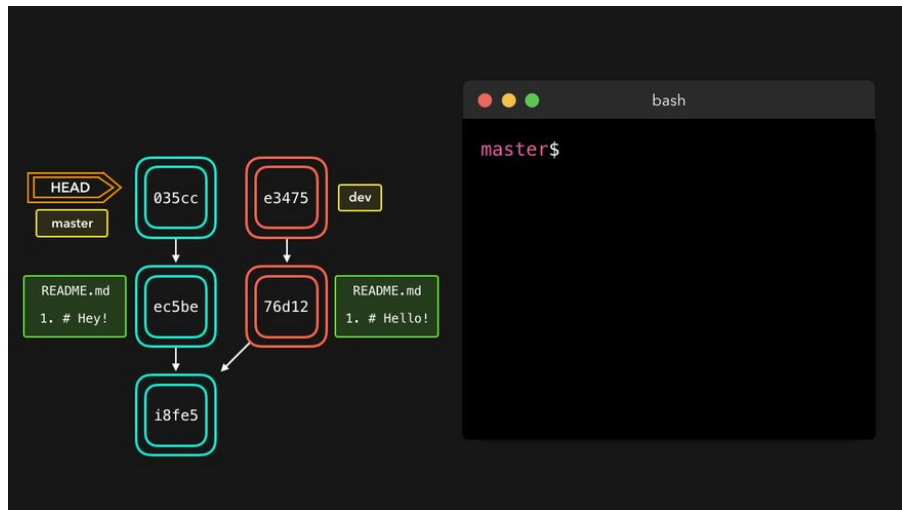


credits: [CS visualized from Lydia Hallie](#)

Tutorial -- Merge with Conflict

- If two branches we trying to merge both modified the same part of the file
 - Git could not figure out which part (version) to use
 - we need to handle such situation manually
- Open conflicted file and
 - reserved the part you desired
 - deleted the others

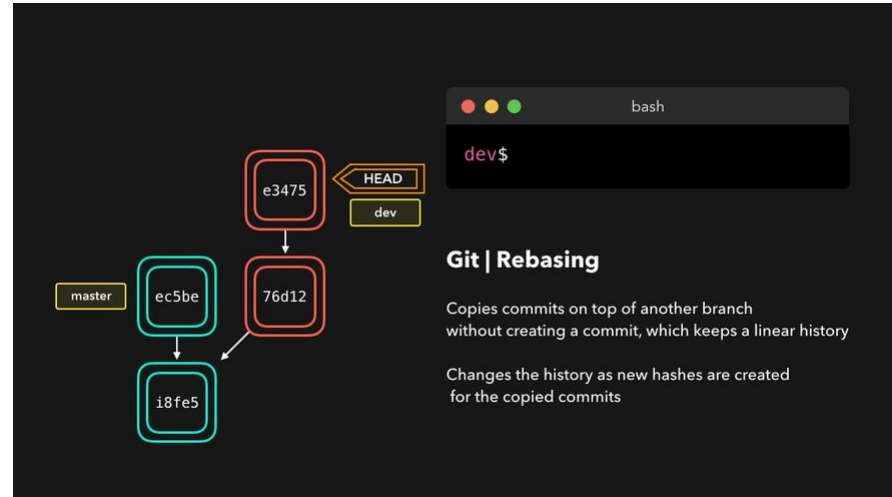
```
here is some content not affected by the conflict
<<<<<< main
this is conflicted text from main
=====
this is conflicted text from feature branch
>>>>>> feature branch;
```



credits: [CS visualized from Lydia Hallie](#)

Tutorial -- Rebase

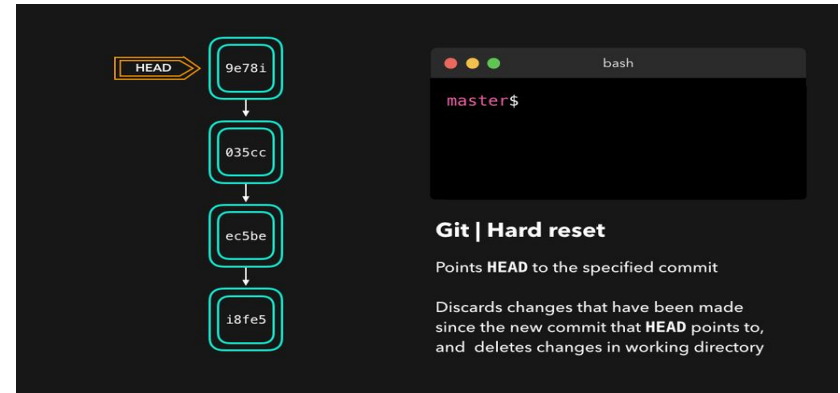
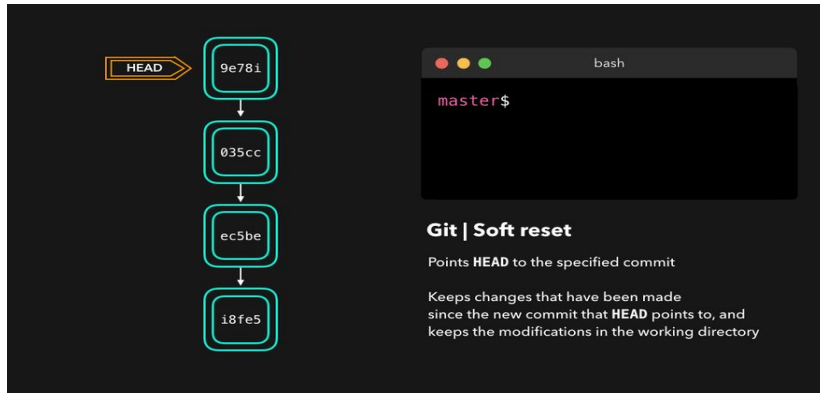
- While **git merge [target branch]** combines two branches together, the command would also reserve the target branch
 - it would be chaos when there're too many branches
- Command **git rebase [target branch]** provides us other option
 - duplicate commits in target branch to current branch
 - delete commits in target branch



credits: [CS visualized from Lydia Hallie](#)

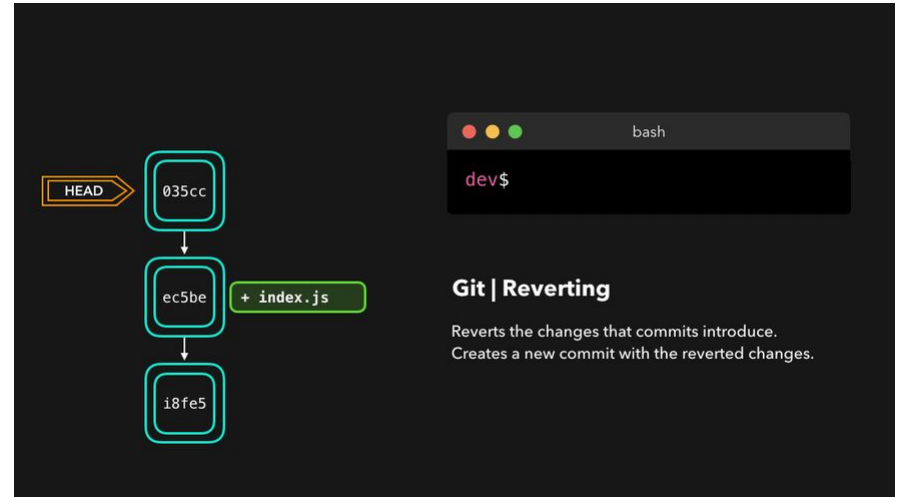
Tutorial -- Reset

- **git reset** is like a undo operation, it can get rids of all current staged files and gives us control over HEAD should point to
- There're several types of reset, all of them move our HEAD pointer ot specified commit



Tutorial -- Revert

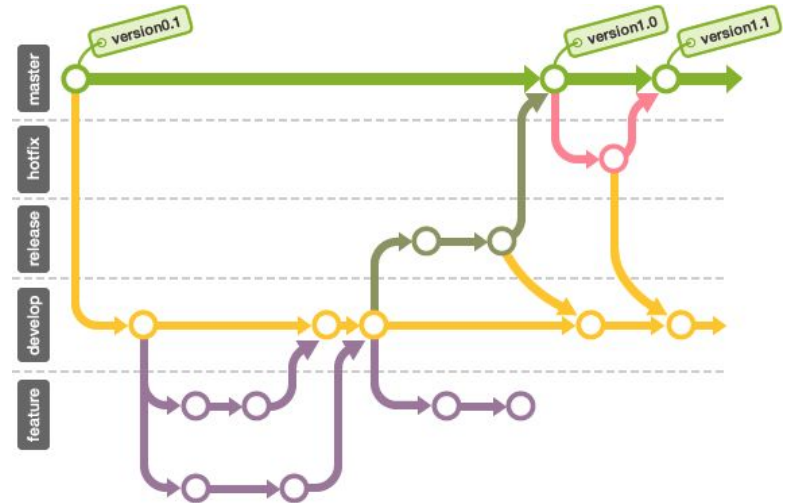
- Another way to undo changes is performing **git revert**
- Reverting a certain commit would create a new commit that contains the reverted changes



credits: [CS visualized from Lydia Hallie](#)

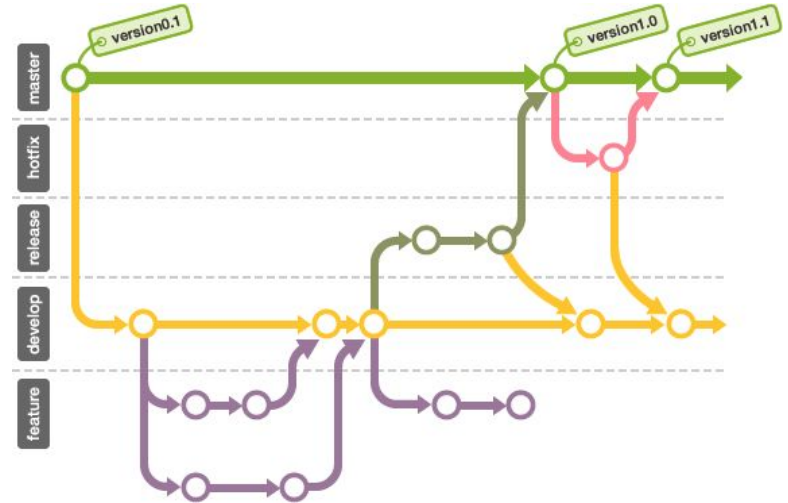
Tutorial -- Lightweight Tag

- Tags are references that point to specific commits in Git history
 - used for a marked version release (i.e. v2.1.0)
 - **git tag <tagname>** would create a lightweight tags which make linking to relevant commits more convenience
 - otherwise we have to check and copy hashes of the commits



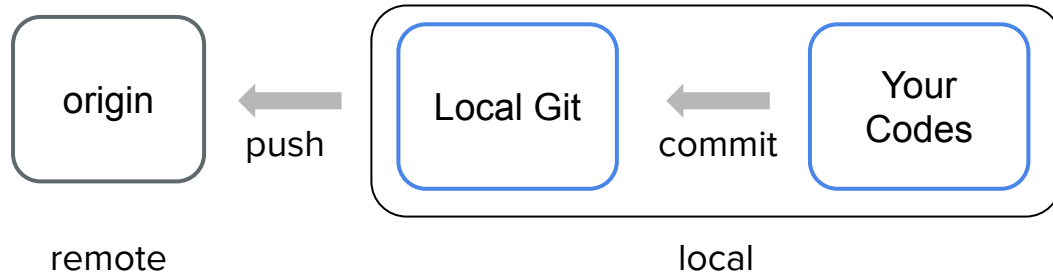
Tutorial -- Annotated Tag

- Compared to lightweight tags, annotated tags have more information, which is called metadata
 - metadata mainly stores the tagger name, email and date
 - annotated tags require more securities
 - most of the annotated tags are use for public release
 - **git tag -a <tagname>**



Tutorial -- Push codes to Origin (remote)

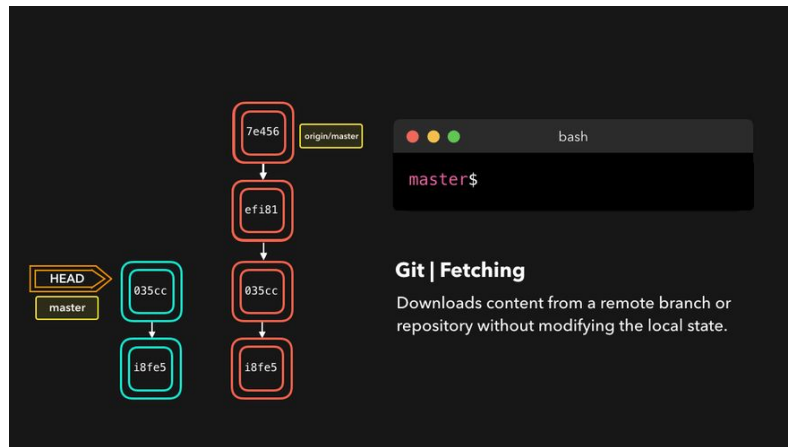
- After adding codes at local, you commit and push them to origin



```
ubuntu ~/Downloads/git_test main >1 git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 344 bytes | 344.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:ander1119/git_test.git
13d3b48..72d6016 main -> main
```

Tutorial -- Synchronize Codes from Origin (remote)

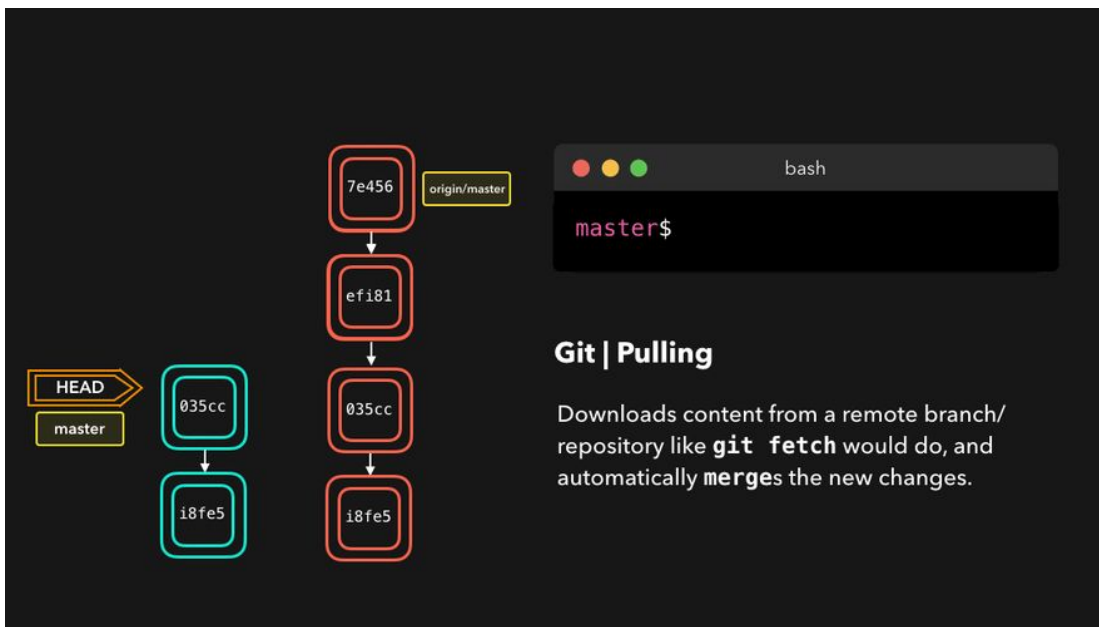
- It's important to stay up to date for all team members while working as a team on a project
 - in case of you and your teammates fix the same bug or develop the same feature simultaneously
 - to get the most recent change, use **git fetch** to update repo
 - after calling git fetch, one should call **git merge [remote branch]** to merge current branch with remote branch



credits: [CS visualized from Lydia Hallie](#)

Tutorial -- Synchronize Codes from Origin (remote)

- Alternatively, users can choose to simply use **git pull** instead of combination of **git fetch**; **git merge**



credits: [CS visualized from Lydia Hallie](#)

Other Useful Tools

- `git status [-uno]`
 - to show status of your changing files
- `git diff [filename]`
 - to know what you modified with [filename]
- `git diff [commit hash 1] [commit hash 2] [filename]`
 - to know the difference of the same file on two different commits
- `git log [--graph] [--oneline]`
- `git reset [commit hash/HEAD[^]] [--hard]`
 - reset commit to specified
 - “^” the previous commit
 - “--hard” recover commit completely

hw

—

Learn Git Branching

- Complete “Main” and “Remote” tasks in <https://learngitbranching.js.org/> and snapshot
- Single or Multiple images is allowed

Submission Form

- Compress snapshot images with your python assignment
- Your submission should be a .zip file

[student_id].zip

└─ [student_id] /

└─ hw.py

└─ git /

└─ [student_id]-0.png

└─ [student_id]-1.png

└─ ...

r11922109.zip

└─ r11922109 /

└─ hw.py

└─ git /

└─ r11922109-0.png

└─ r11922109-1.png

└─ ...