



# GitHub

TA

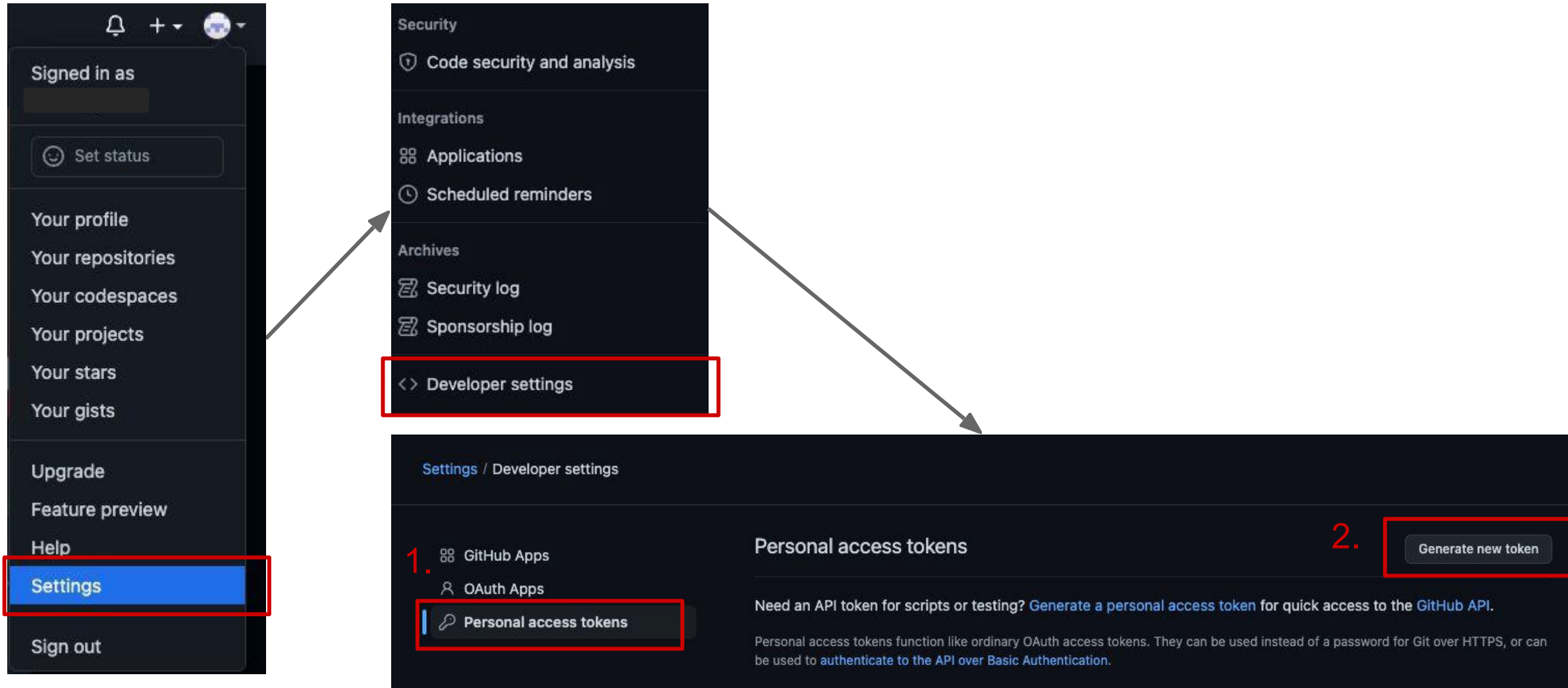
劉亭妤 Joanne



# Notices!!

Before the course begins, please register a GitHub account.

# 1. Get the access token : PAT key



# 1. Get the access token : PAT key

**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**

What's this token for?

**Expiration \***

30 days The token will expire on Sun, Oct 23 2022

**Select scopes**

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

<input checked="" type="checkbox"/> <b>repo</b>	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input checked="" type="checkbox"/> <b>workflow</b>	Update GitHub Action workflows
<input type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input type="checkbox"/> admin:org	Full control of orgs and teams, read and write org projects
<input type="checkbox"/> write:org	Read and write org and team membership, read and write org projects
<input type="checkbox"/> read:org	Read org and team membership, read org projects
<input type="checkbox"/> manage_runners:org	Manage org runners and runner groups
<input type="checkbox"/> admin:public_key	Full control of user public keys

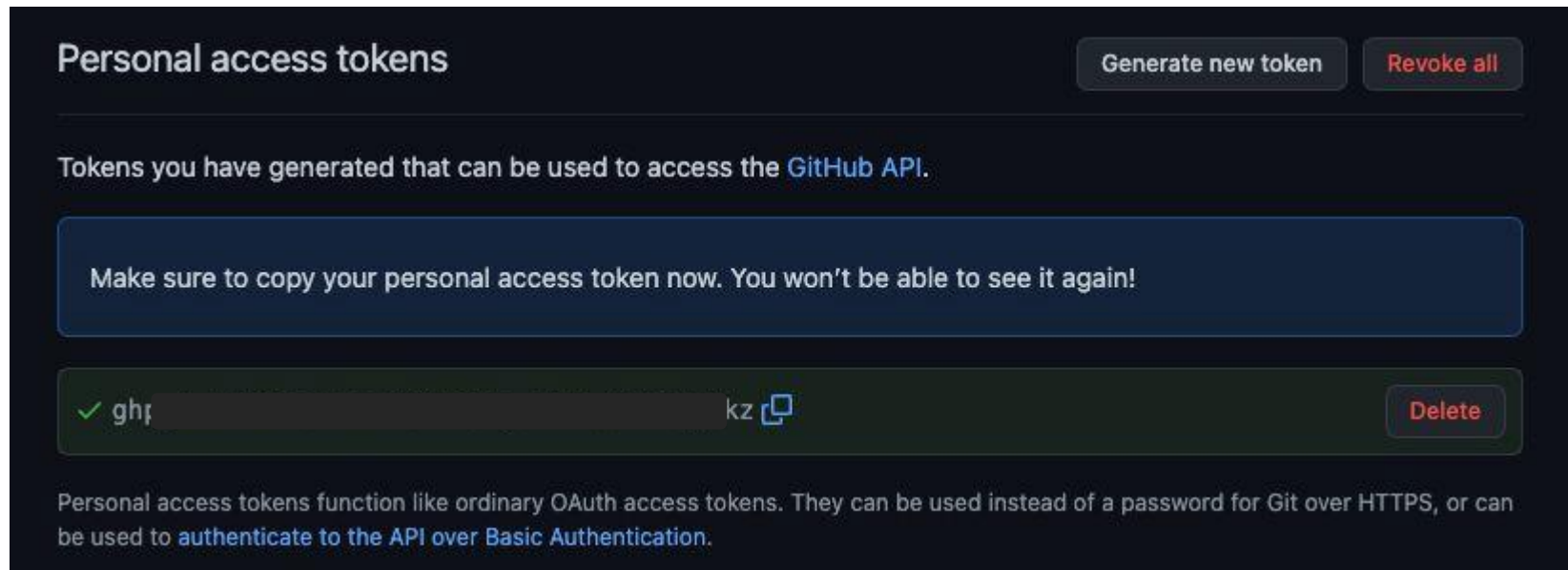
If expired, need to generate again

<input type="checkbox"/> admin:public_key	Full control of user public keys
<input type="checkbox"/> write:public_key	Write user public keys
<input type="checkbox"/> read:public_key	Read user public keys
<input type="checkbox"/> admin:repo_hook	Full control of repository hooks
<input type="checkbox"/> write:repo_hook	Write repository hooks
<input type="checkbox"/> read:repo_hook	Read repository hooks
<input type="checkbox"/> admin:org_hook	Full control of organization hooks
<input type="checkbox"/> gist	Create gists
<input type="checkbox"/> notifications	Access notifications
<input type="checkbox"/> user	Update ALL user data
<input type="checkbox"/> read:user	Read ALL user profile data
<input type="checkbox"/> user:email	Access user email addresses (read-only)
<input type="checkbox"/> user:follow	Follow and unfollow users
<input type="checkbox"/> delete_repo	Delete repositories
<input type="checkbox"/> write:discussion	Read and write team discussions
<input type="checkbox"/> read:discussion	Read team discussions
<input type="checkbox"/> admin:enterprise	Full control of enterprises
<input type="checkbox"/> manage_runners:enterprise	Manage enterprise runners and runner groups
<input type="checkbox"/> manage_billing:enterprise	Read and write enterprise billing data
<input type="checkbox"/> read:enterprise	Read enterprise profile data
<input type="checkbox"/> project	Full control of projects
<input type="checkbox"/> read:project	Read access of projects
<input type="checkbox"/> admin:gpg_key	Full control of public user GPG keys
<input type="checkbox"/> write:gpg_key	Write public user GPG keys
<input type="checkbox"/> read:gpg_key	Read public user GPG keys
<input type="checkbox"/> admin:ssh_signing_key	Full control of public user SSH signing keys
<input type="checkbox"/> write:ssh_signing_key	Write public user SSH signing keys
<input type="checkbox"/> read:ssh_signing_key	Read public user SSH signing keys

**Generate token** Cancel

# 1. Get the access token : PAT key

- Make sure to copy your access token to your note!
- This will use when you want to push code to GitHub.
- (let environment remember your key) git config --global credential.helper manager



# 1. Get the access token : SSH key

- terminal
  - `ssh-keygen -t rsa -b 4096 -C "your_email@example.com"`
  - Enter your passphrase twice

```
Generating public/private rsa key pair.  
Enter a file in which to save the key (/Users/you/.ssh/id_rsa): [Press enter]  
Enter passphrase (empty for no passphrase): [Type a passphrase]  
Enter same passphrase again: [Type passphrase again]
```

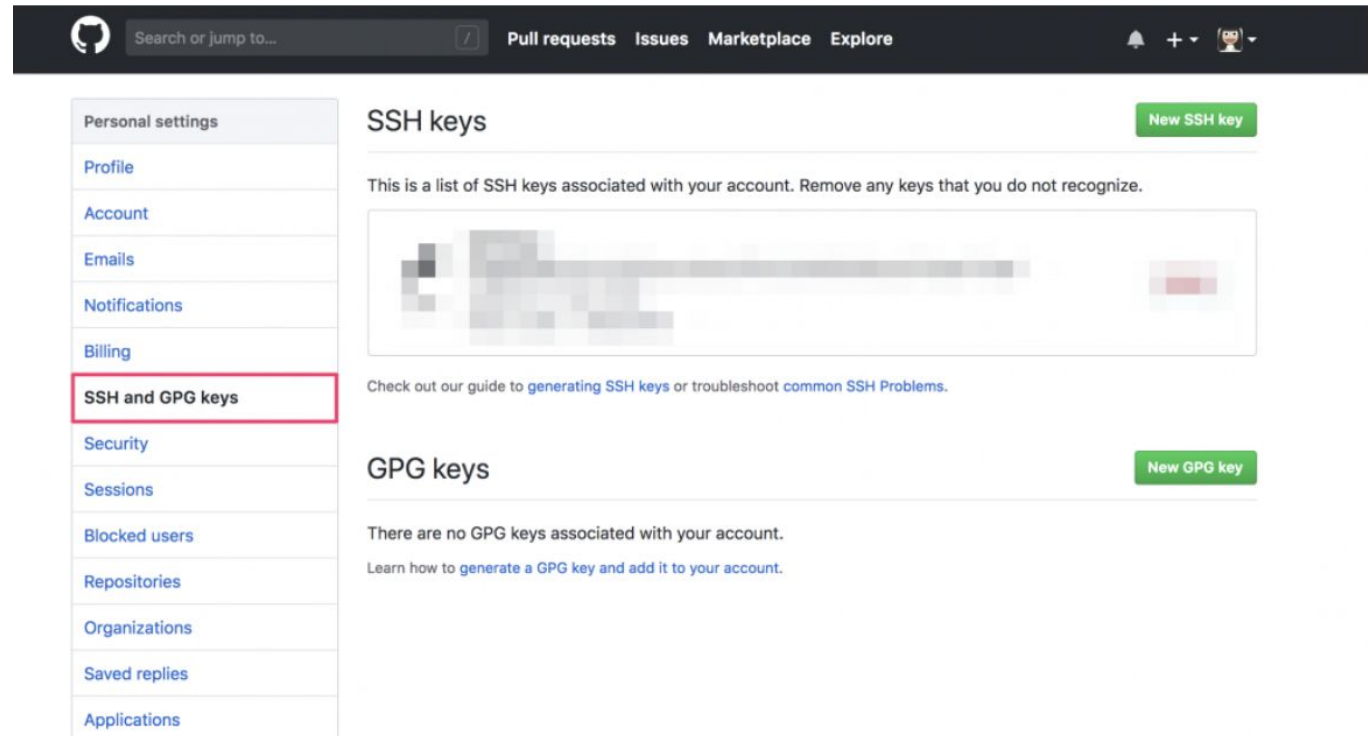
# 1. Get the access token : SSH key

- terminal
  - (generate ssh key) `ssh-keygen -t rsa -b 4096 -C "your_email@example.com"`
  - Enter your passphrase twice
  - (open ssh agent) `eval "$(ssh-agent -s)"`
  - (add key to agent) `ssh-add -K ~/.ssh/id_rsa`

```
Generating public/private rsa key pair.  
Enter a file in which to save the key (/Users/you/.ssh/id_r  
sa): [Press enter]  
Enter passphrase (empty for no passphrase): [Type a passphr  
ase]  
Enter same passphrase again: [Type passphrase again]
```

# 1. Get the access token : SSH key

- terminal
  - (show ssh key) `cat ~/.ssh/id_rsa.pub`
  - copy the key
- GitHub
  - register key






## 2. Create a Project : from Remote

- Configure user information via VScode terminal
  - `git config --global user.name "Your Name"`
  - `git config --global user.email "you@example.com"`
  - (check configuration) `git config --global --list`
- Login GitHub (<https://github.com/>)
- New a Repository
  - SSH : `git clone git@github.com:{user.name}/{repo.name}.git`
  - HTTPS : `git clone https://github.com/{user.name}/{repo.name}.git`

**Create a new repository**  
Repositories contain a project's files and version history. Have a project elsewhere? [Import a repository](#).  
Required fields are marked with an asterisk (\*).

**1 General**

Owner \*  folivora-hi


**Repository name \***


Great repository names are short and memorable. How about **fantastic-disco**?

**Description**

0 / 350 characters

**2 Configuration**

**Choose visibility \***  
Choose who can see and commit to this repository  **Public**

**Add README**  
READMEs can be used as longer descriptions. [About READMEs](#) Off 

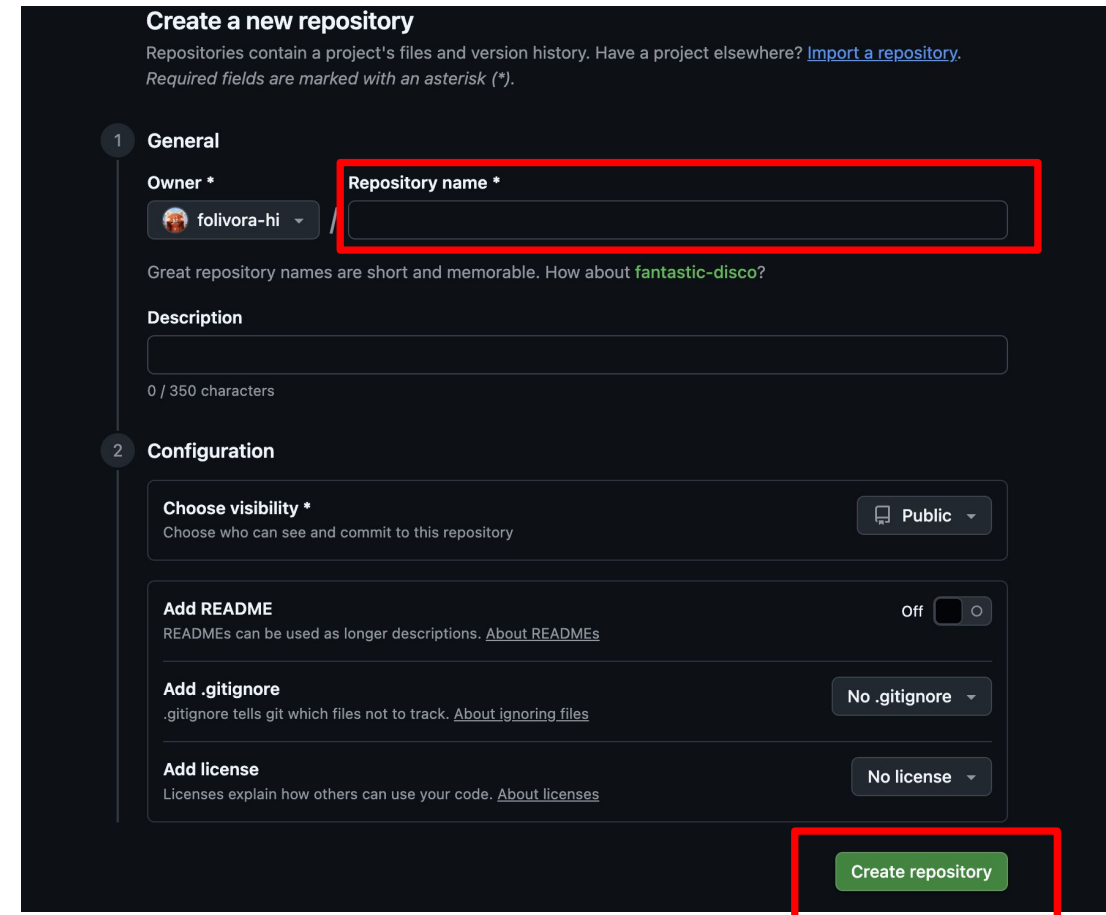
**Add .gitignore**  
.gitignore tells git which files not to track. [About ignoring files](#) **No .gitignore**

**Add license**  
Licenses explain how others can use your code. [About licenses](#) **No license**

**Create repository**

## 2. Create a Project : from Local

- Configure user information via VScode terminal
  - `git config --global user.name "Your Name"`
  - `git config --global user.email "you@example.com"`
  - (check configuration) `git config --global --list`
  - `git init`
- Login GitHub (<https://github.com/>)
- New a Repository
  - `git remote add origin`  
`git@github.com:{user.name}/{repo.name}.git`
  - `git branch -M main`
  - `git push -u origin main`



**Create a new repository**

Repositories contain a project's files and version history. Have a project elsewhere? [Import a repository](#).  
Required fields are marked with an asterisk (\*).

1 **General**

Owner \* folivora-hi

**Repository name \***

Great repository names are short and memorable. How about `fantastic-disco`?

**Description**

0 / 350 characters

2 **Configuration**

**Choose visibility \***  
Choose who can see and commit to this repository Public

**Add README**  
READMEs can be used as longer descriptions. [About READMEs](#) Off

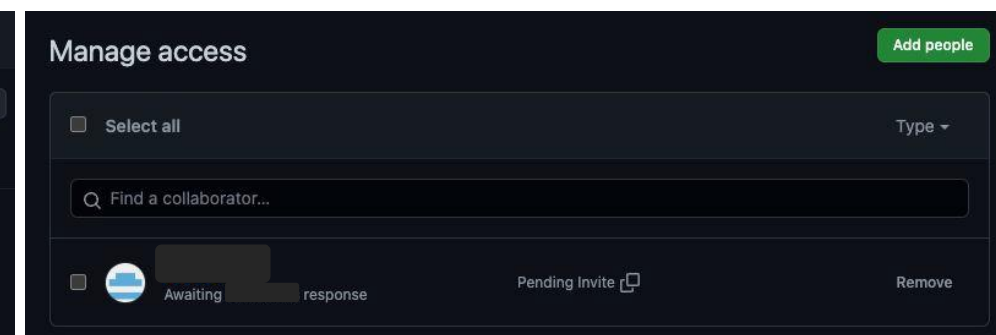
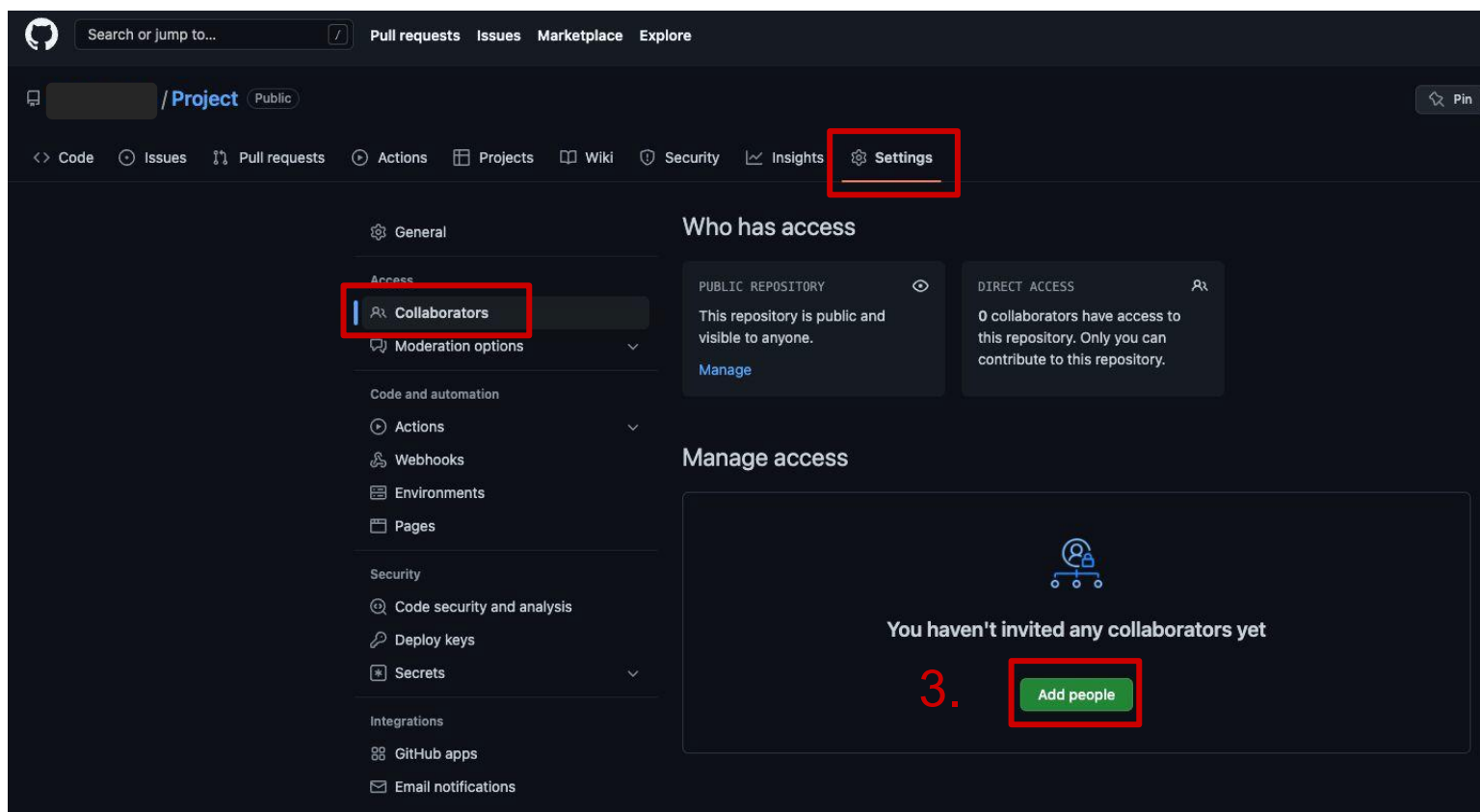
**Add .gitignore**  
.gitignore tells git which files not to track. [About ignoring files](#) No .gitignore

**Add license**  
Licenses explain how others can use your code. [About licenses](#) No license

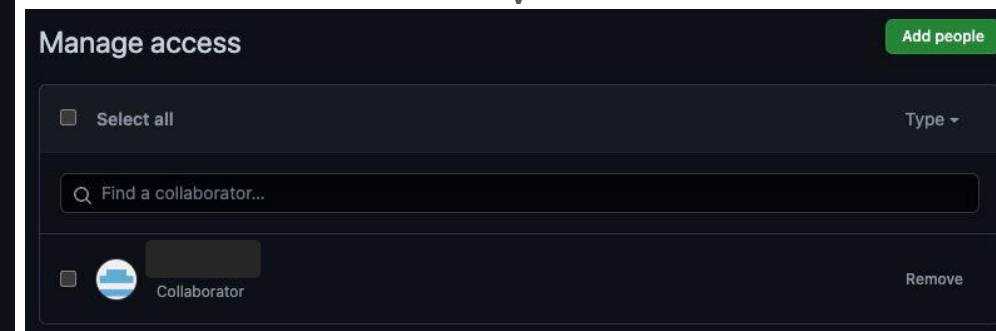
**Create repository**

## 2. Create a Project

- Add your members to repository



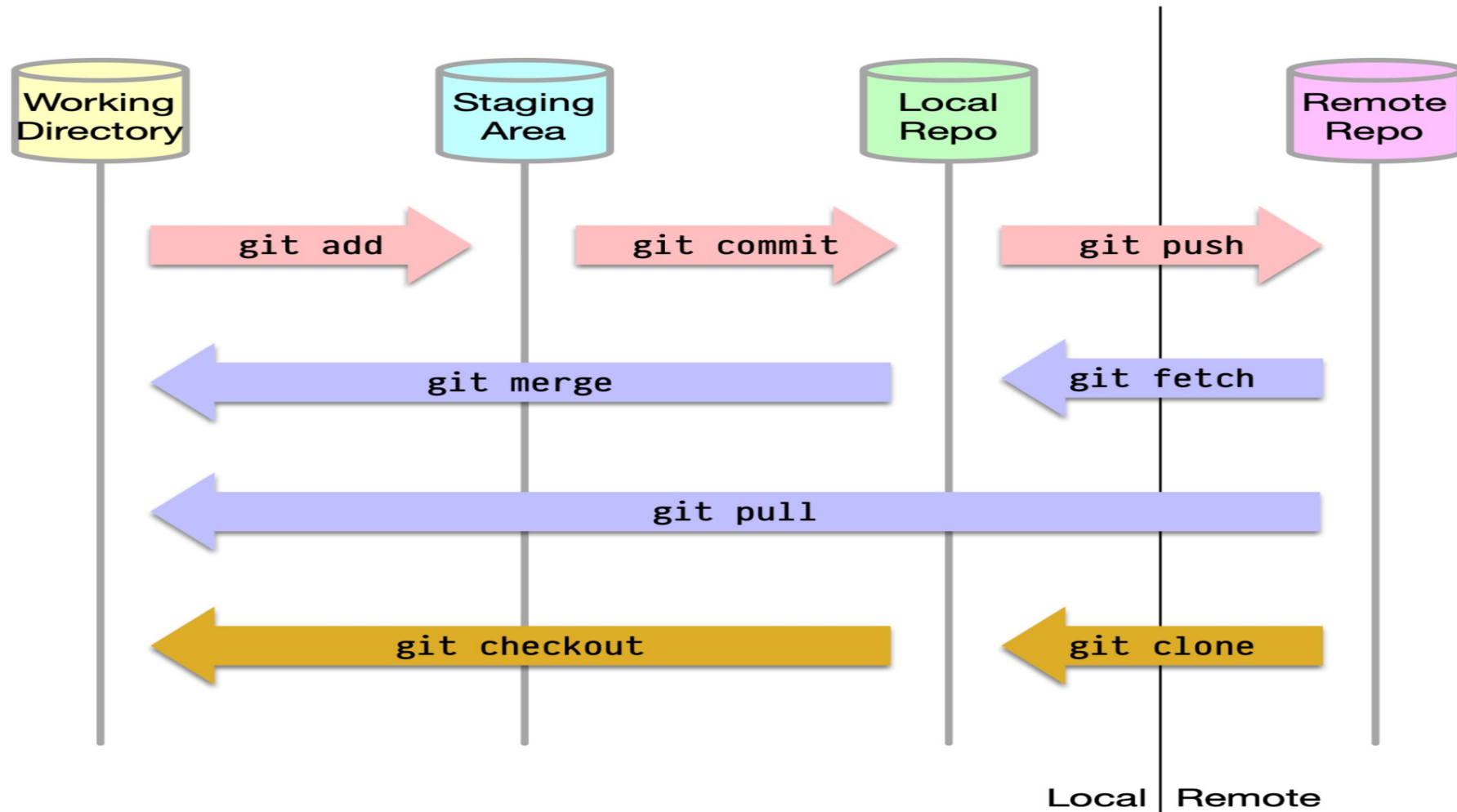
After your  
members join



# 3. Git Command : Basic

How Git Commands work

ByteByteGo.com



# 3. Git Collaboration

- Always work on a branch.
- Keep main synced with remote.
  - Always `git pull origin main` before creating a new branch to ensure it is based on the latest code.
- Clear Commit Messages.
  - Avoid vague messages like "fix bug." Be descriptive, e.g., `fix: resolve header alignment issue on mobile`.
- Don't Upload Unnecessary Files.
  - Use `.gitignore` to exclude compiled files, temp files, and environment configs.
- Merge via Pull Requests (with reviews).

# 3. Git Collaboration : Branch

Never develop directly on main. Always create a new branch for features or fixes, e.g., feature/login or bugfix/header.

Command	Description
<code>git branch</code>	List all local branches ( * marks the current branch).
<code>git branch -r</code>	List all remote branches.
<code>git branch -a</code>	List all local and remote branches.
<code>git branch &lt;name&gt;</code>	Create a new branch (but stay on the current one).
<code>git checkout &lt;name&gt;</code>	Switch to the branch <code>&lt;name&gt;</code> .
<code>git switch &lt;name&gt;</code>	Newer alternative to switch to branch <code>&lt;name&gt;</code> .
<code>git checkout -b &lt;name&gt;</code>	Create and switch to a new branch in one step.
<code>git switch -c &lt;name&gt;</code>	Newer alternative to create and switch to a new branch.
<code>git checkout main</code> <code>git merge &lt;name&gt;</code>	Merge branch <code>&lt;name&gt;</code> into <code>main</code> .
<code>git branch -d &lt;name&gt;</code>	Delete a local branch (only if merged).
<code>git branch -D &lt;name&gt;</code>	Force delete a local branch (even if not merged).
<code>git push -u origin &lt;name&gt;</code>	Push local branch <code>&lt;name&gt;</code> to remote and set upstream tracking.
<code>git push origin --delete &lt;name&gt;</code>	Delete a remote branch <code>&lt;name&gt;</code> .

# 3. Git Collaboration : Merge & Conflict

**Don't merge directly. Use Pull Requests, get code reviewed, and then merge into main.**

[Link] <https://ithelp.ithome.com.tw/articles/10339487>

**When merge conflicts occur, don't force push. Resolve locally, test, then push again.**

[Link] <https://heidiliu2020.github.io/git-github/>

# Notice

- Send your GitHub link and contact information via Google Sheets!

[\[Link\]](#)



# Supplement

- Basic Git & GitHub Concept : <https://www.youtube.com/watch?v=FKXRiAiQFiY>
- Git Stash (暫存): [https://www.maxlist.xyz/2018/11/02/git\\_tutorial/](https://www.maxlist.xyz/2018/11/02/git_tutorial/)
- [https://www.youtube.com/watch?v=VShhhq\\_5sMc&list=PLBd8JGCAcUAF2\\_im\\_kk\\_ZTfEAKnlmfPJy](https://www.youtube.com/watch?v=VShhhq_5sMc&list=PLBd8JGCAcUAF2_im_kk_ZTfEAKnlmfPJy)

