
Xi'an Jiaotong-Liverpool University
西交利物浦大學

# *Git(Hub) Tutorial*

Chengtao Ji

(Materials from Online)

2025/3/10

# Content



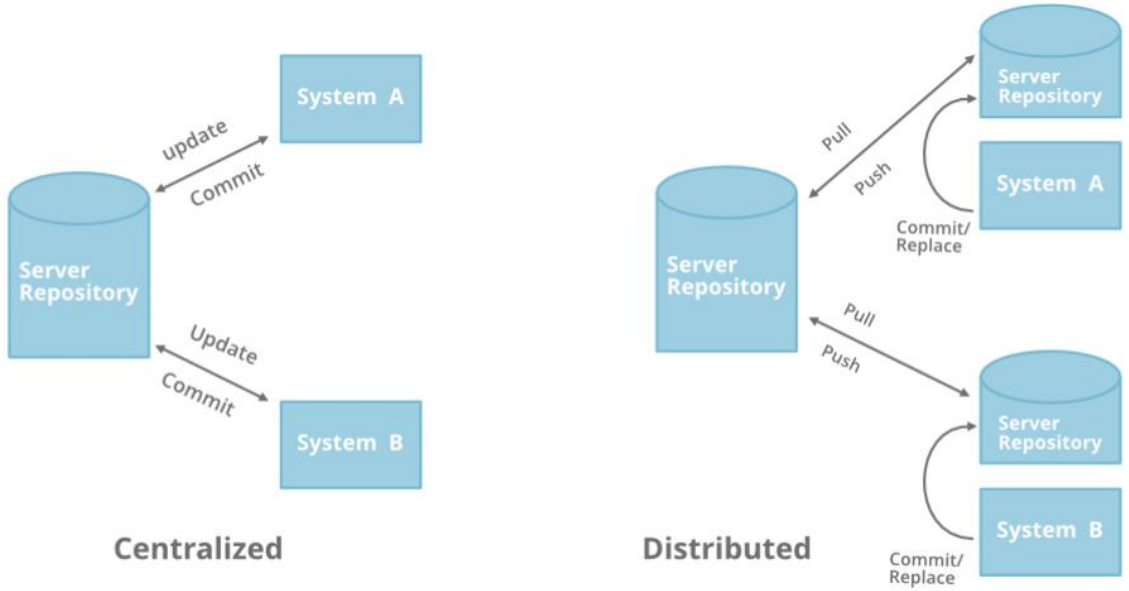
- ❑ Version Control System
- ❑ Git
- ❑ GitHub

# Challenge

- **Collaboration:** There are so many people located in different places that there may be a need to communicate for a particular reason, or a set of people are working on the same project but are from other regions.
- **Storing Versions:** The project is completed into several versions; in that situation, keeping all such commits in a single place is a considerable challenge.
- **Restoring Previous Versions:** Sometimes, there is a need to go back to the earlier versions and the bug's root cause.
- **Figure Out What Happened:** It is critical to know what changes were made to the previous versions of the source code or where exactly the changes have been made in a file.
- **Backup:** If the user's system or disk breaks down and there is no backup, all efforts go in vain.

# Version Control System

Version control systems are a category of software tools that helps in recording changes made to files by keeping a track of modifications done to the code.




- **Centralized version control systems** contain just one server repository; each user gets a working copy. You need to commit to reflecting your changes in the repository. Others can see your changes by updating.

- **Distributed version control systems** contain multiple repositories. Each user has their repository and working copy. Just committing to your changes will not give others access to them.

# Benefits

- **Multiple people can work simultaneously on a single project.** Everyone works on and edits their copy of the files, and it is up to them when they wish to share the changes they made with the rest of the team.
- **It enables one person to use multiple computers to work on a project,** which is valuable even if you are working alone.
- **It integrates the work that is done simultaneously by different members of the team.** In some rare cases, when two people make conflicting edits to the same line of a file, then human assistance is requested by the version control system to decide what should be done.
- **It provides access to the historical versions of a project.** This is insurance against computer crashes or data loss. You can easily roll back to a previous version if any mistake is made. It is also possible to undo specific edits without losing the work done in the meantime. It can be easily known when, why, and by whom any part of a file was edited.

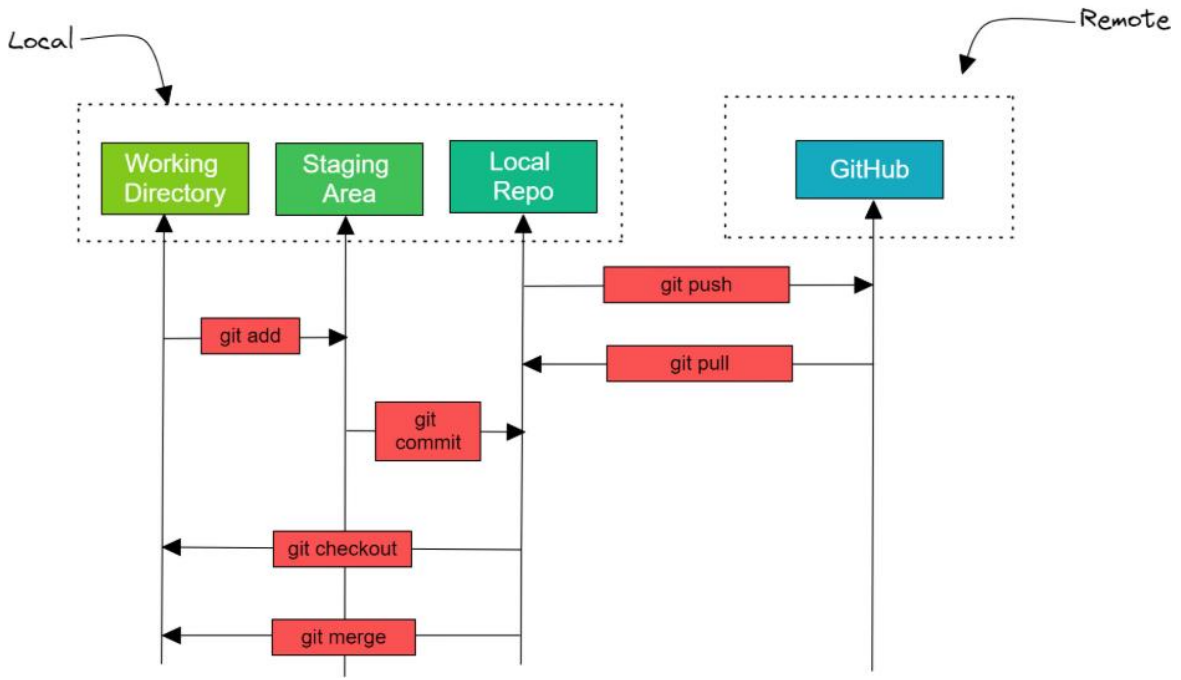
# Git

**Git** is a version control system for tracking computer file changes. It is generally used for source code management in software development.

- **Tracks history**
- **Free and open source**
- **Supports non-linear development**
- **Creates backups**
- **Scalable**
- **Supports collaboration**
- **Branching is easier**
- **Distributed development**

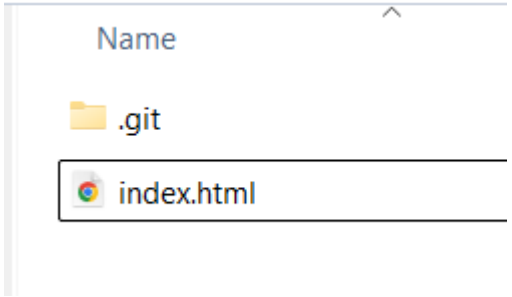
# Git Repository Structure & Workflow

- **Working directory:** This is your local directory where you **make the project (write code) and make changes**.
- **Staging Area:** this is an area where you **first need to put your project before committing**. Other team members use this for code review.
- **Local Repository:** this is your **local repository where you commit changes to the project before pushing them to the central repository** on GitHub. This is what is provided by a distributed version control system. This corresponds to the .git folder in our directory.
- **Central Repository:** This is the **main project on the central server**, a copy of which is with every team member as a local repository.

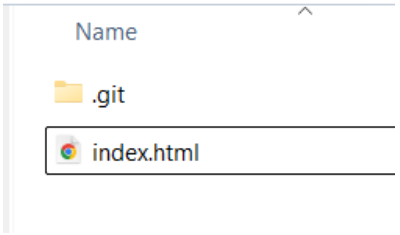


# Create a local repository

1. **Download and install Git**: https://git-scm.com/downloads
2. **Create a folder and Navigate into that folder:** *mkdir myproj && cd myproj*
3. **Configure Git:**
   1. *git config (--global) user.name "yourname";*
   2. *git config (--global) user.email "youremail@email.com"*
4. **Initialize the local repository:** *git init* (initializes your directory to work with git and makes a local repository)
5. **Add new files:** *code index.html*
6. **Check the status:** *git status*


# Create a local repository

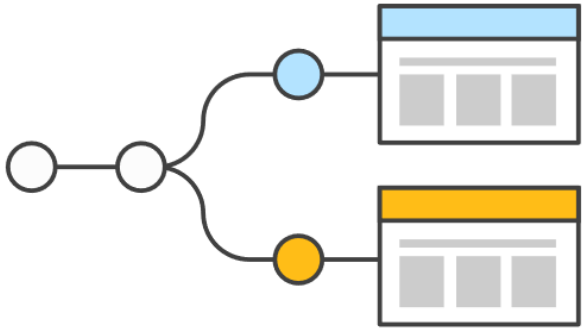



1. **Add the new document to the Staging Environment:** *git add index.html*
2. **Make a commit:** *git commit –m "description"*
3. **View commit history:** *git log*
4. **Help:** *git "command" -–help OR git help --all*

```
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git add index.html
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git status
On branch master

No commits yet

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

PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git commit -m "commit an index page"
[master (root-commit) 71293e2] commit an index page
 1 file changed, 200 insertions(+)
 create mode 100644 index.html
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git status
On branch master
nothing to commit, working tree clean
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git log
commit 71293e21eefbed5efa3cbeadc167422df1133ad2 (HEAD -> master)
Author: ChengtaoJi <JiChengtao@gmail.com>
Date:   Thu Mar 6 18:50:22 2025 +0800

    commit an index page
```

# Git Branch

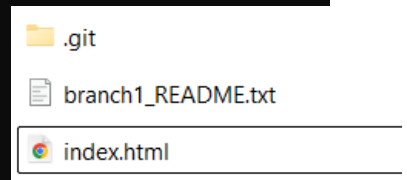
A branch is a **separate version of the main repository**. Branches allow you to **work on different parts of a project** without impacting the main branch.



1. **Create a new branch**: *git branch new_branch_name*
2. **List all branches:** *git branch*
3. **Switch to a branch:** git checkout branch_name
4. **Delete a branch:** *git branch –d branch_name*

```
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git status
On branch master
nothing to commit, working tree clean
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git branch branch1
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git branch
  branch1
* master
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git branch branch2
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git branch
  branch1
  branch2
* master
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git checkout branch1
Switched to branch 'branch1'
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git branch
* branch1
  branch2
  master
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git branch -d branch2
Deleted branch branch2 (was 71293e2).
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git branch
* branch1
  master
```

```
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git checkout branch1
Already on 'branch1'
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git branch
* branch1
  branch2
  master
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> ls


    Directory: C:\Users\Chengtao\Desktop\GitDemo\myproj


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----          3/6/2025   6:39 PM           5546 index.html


PS C:\Users\Chengtao\Desktop\GitDemo\myproj> code branch1_README.txt
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git add branch1_README.txt
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git commit -m "add branch1_README.txt"
[branch1 47489f0] add branch1_README.txt
 1 file changed, 1 insertion(+)
 create mode 100644 branch1_README.txt
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git status
On branch branch1
nothing to commit, working tree clean
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> ls


    Directory: C:\Users\Chengtao\Desktop\GitDemo\myproj


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----          3/8/2025   4:36 PM             23 branch1_README.txt
-a----          3/6/2025   6:39 PM           5546 index.html


PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git checkout branch2
Switched to branch 'branch2'
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git branch
  branch1
* branch2
  master
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> ls


    Directory: C:\Users\Chengtao\Desktop\GitDemo\myproj


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----          3/6/2025   6:39 PM           5546 index.html
```




# Git diff

Diffing is a function that **takes two input data sets** and **outputs the changes between them.**

*git diff* is a multi-use Git command that, when executed, runs a diff function on Git data sources. These data sources can be commits, branches, files, and more.

**Comparing two branches:** git diff branch1 (branch2)

**1. Comparison input:** *diff --git a/branch1_README.txt b/branch1_README.txt*

This line displays the input sources of the diff. We can see that *a/branch1_README.txt* and *b/branch1_README.txt* have been passed to the diff.

2**. Metadata:**

*deleted file mode 100644*

*index 2c536dd..0000000*

This line displays some internal Git metadata. You will most likely not need this information. The first one indicates that the file branch1_README.txt has been deleted in branch2. The numbers in this output correspond to Git object version hash identifiers.

**3. Markers for changes**:

*--- a/branch1_README.txt*

*+++ /dev/null*

These lines are a legend that assigns symbols to each diff input source: The file in branch1 && The file is missing in branch2 (it has been deleted).

**4. Diff chunks:**

*@@ -1 +0,0 @@*

*-Hi, you are in branch1.*

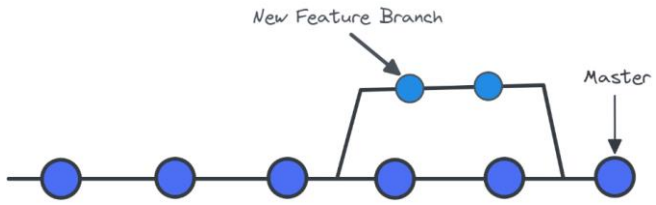
@@ -1 +0,0 @@: This is a diff header indicating that line 1 in branch1 is being compared to a non-existent line in branch2. Each chunk is prepended by a header enclosed within @@ symbols. The content of the header is a summary of changes made to the file.

-Hi, you are in branch1.: This line is present in branch1 but missing in branch2.

\ No newline at end of file: This indicates that the file in branch1 does not have a newline character at the end.

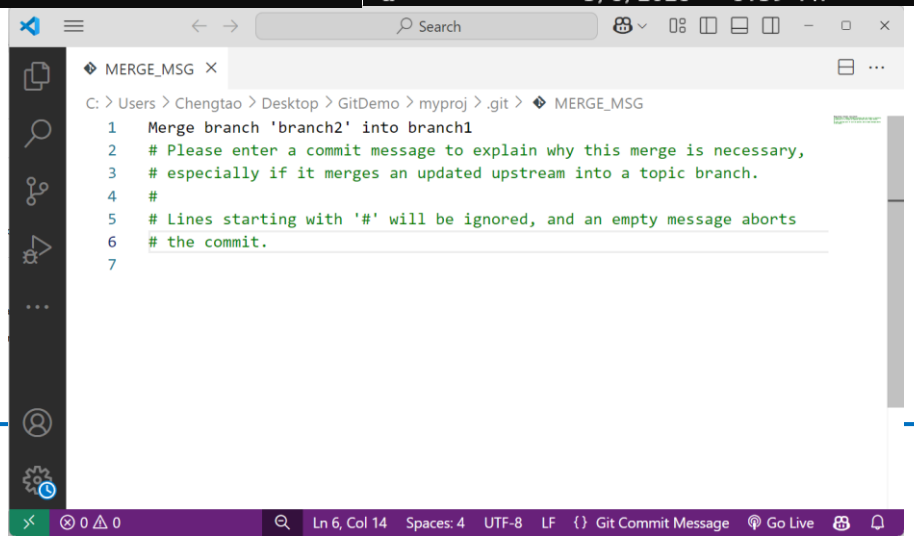
```
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git branch
  branch1
* branch2
  master
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> ls


    Directory: C:\Users\Chengtao\Desktop\GitDemo\myproj


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----          3/6/2025   6:39 PM           5546 index.html


PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git diff branch1
diff --git a/branch1_README.txt b/branch1_README.txt
deleted file mode 100644
index 2c536dd..0000000
--- a/branch1_README.txt
+++ /dev/null
@@ -1 +0,0 @@
-Hi, you are in branch1.
\ No newline at end of file
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git branch
  branch1
* branch2
  master
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git diff branch1 branch2
diff --git a/branch1_README.txt b/branch1_README.txt
deleted file mode 100644
index 2c536dd..0000000
--- a/branch1_README.txt
+++ /dev/null
@@ -1 +0,0 @@
-Hi, you are in branch1.
\ No newline at end of file
```

# Git Merge




```
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git checkout branch1
Switched to branch 'branch1'
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> ls


    Directory: C:\Users\Chengtao\Desktop\GitDemo\myproj


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----          3/9/2025   9:54 AM             23 branch1_README.txt
-a----          3/6/2025   6:39 PM           5546 index.html


PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git checkout branch2
Switched to branch 'branch2'
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> code branch2_README.txt
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git add --all
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git commit -m "branch2 reademe"
[branch2 164507a] branch2 reademe
 1 file changed, 1 insertion(+)
 create mode 100644 branch2_README.txt
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> ls


    Directory: C:\Users\Chengtao\Desktop\GitDemo\myproj


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----          3/9/2025   9:54 AM             23 branch1_README.txt
-a----          3/9/2025   9:54 AM             23 branch2_README.txt
-a----          3/6/2025   6:39 PM           5546 index.html
```

```
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git checkout branch1
Switched to branch 'branch1'
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> ls


    Directory: C:\Users\Chengtao\Desktop\GitDemo\myproj


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----          3/9/2025   9:55 AM             23 branch1_README.txt
-a----          3/6/2025   6:39 PM           5546 index.html


PS C:\Users\Chengtao\Desktop\GitDemo\myproj> git merge branch2
Merge made by the 'ort' strategy.
 branch2_README.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 branch2_README.txt
PS C:\Users\Chengtao\Desktop\GitDemo\myproj> ls


    Directory: C:\Users\Chengtao\Desktop\GitDemo\myproj


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----          3/9/2025   9:55 AM             23 branch1_README.txt
-a----          3/9/2025   9:55 AM             23 branch2_README.txt
-a----          3/6/2025   6:39 PM           5546 index.html
```




1. **Merge the current branch with another one:** git merge branch_name
2. **Make a new commit:** git add files_modified

# Merge Conflicts

- Update the repository frequently to avoid big conflicts.
- Have fewer developers working off the same branch.

# GitHub

- GitHub is a web-based hosting service for Git repositories. It offers all of Git's DVCS SCM and has some additional features. This includes **collaboration functionality** like project management, support ticket management, and bug tracking. With GitHub, developers can share their repositories, access other developers' repositories, and store remote copies of repositories to serve as backups.

- Git is a free, open-source software distributed version control system (DVCS) that manages all source code history. It can keep a history of commits, reverse changes, and let developers share code. To collaborate, each developer must have Git installed on his or her local device.

- If you only want to keep track of your code locally, you don't need to use GitHub. However, if you're going to work with a team, you can use GitHub to modify the project's code collaboratively.

**https://github.com/**

# Generating a new SSH key and adding it to the ssh-agent

1.Open Git Bash and paste the text below, replacing the email used in the example with your GitHub email address.

*ssh-keygen -t ed25519 -C your_email@example.com*

2. In a new ***admin*** *elevated* PowerShell window, ensure the ssh-agent is running. Start it manually with the following example:

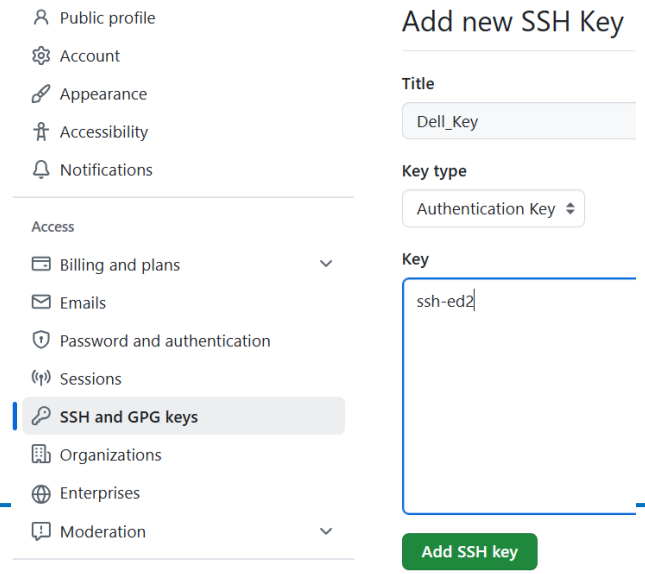
*# start the ssh-agent in the background*
*Get-Service -Name ssh-agent | Set-Service -StartupType Manual*
*Start-Service ssh-agent*

```
PS C:\WINDOWS\system32> Get-Service -Name ssh-agent

Status   Name               DisplayName
------   ----               -----------
Stopped  ssh-agent          OpenSSH Authentication Agent


PS C:\WINDOWS\system32> Set-Service -StartupType Manual

cmdlet Set-Service at command pipeline position 1
Supply values for the following parameters:
Name: ssh-agent
PS C:\WINDOWS\system32> Start-Service ssh-agent
```

```
PS C:\Users\Chengtao\Desktop\GitDemo> ssh-add C:/Users/Chengtao/.ssh/id_ed25519
Identity added: C:/Users/Chengtao/.ssh/id_ed25519 (Chengtao.Ji@xjtlu.edu.cn)
PS C:\Users\Chengtao\Desktop\GitDemo>
```

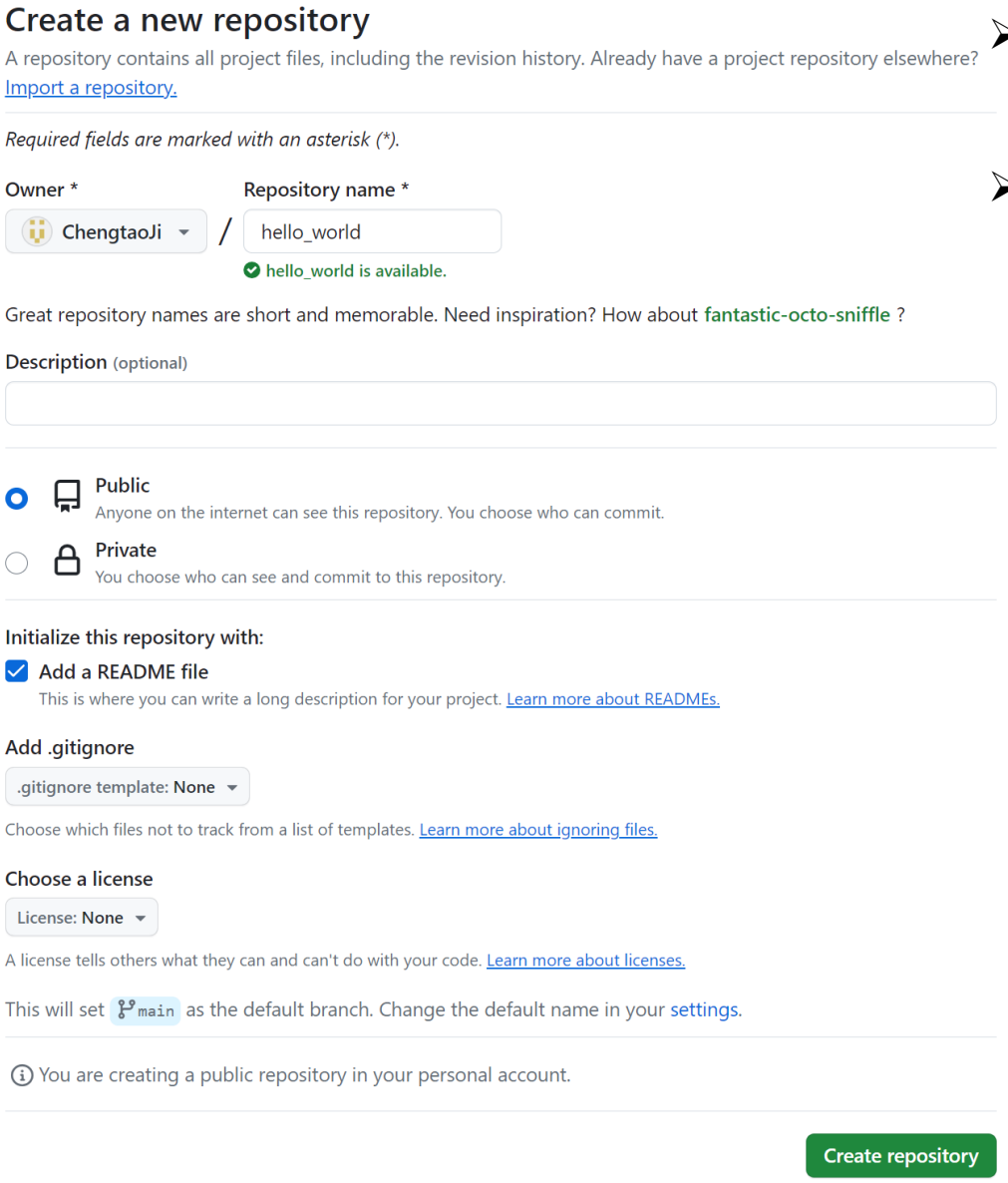
3.Add the SSH public key to your account on GitHub.

*1.Copy the SSH public key to your clipboard.*
*2.In the upper-right corner of any page on GitHub, click your profile photo, then click  Settings.*
*3.In the "Access" section of the sidebar, click  SSH and GPG keys.*
*4.Click New SSH key or Add SSH key.*
*5.In the "Title" field, add a descriptive label for the new key. For example, if you're using a personal laptop, you might call this key "Dell_Key."*
*6.Select the type of key, either authentication or signing.*
*7.In the "Key" field, paste your public key.*
*8.Click Add SSH key.*

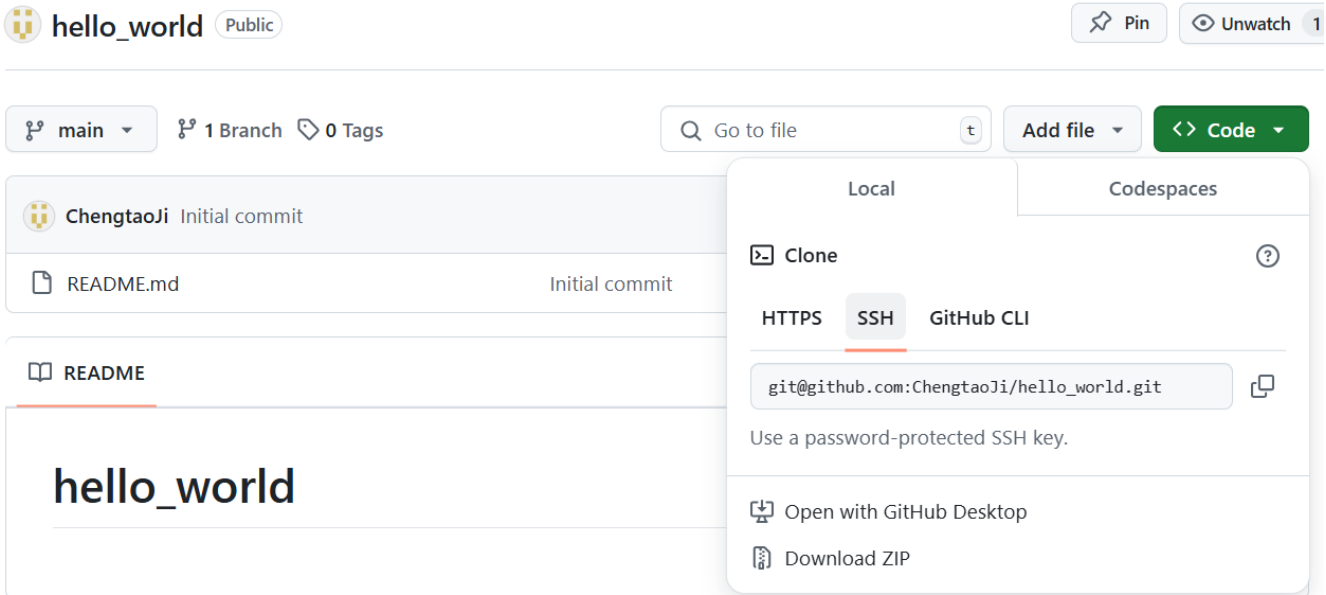



*https://docs.github.com/en/authentication/connecting-to-github-with-ssh/*generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent

# Create a new repository on GitHub




- To create a new repo on GitHub, log in and go to the GitHub home page. You can find the “New repository” option under the “+” sign next to your profile picture in the top right corner of the navbar.
- When you're done filling out the information, press the 'Create repository' button to make your new repo.



# Link local Git repository to GitHub repository

1. **Create a repository on GitHub with the name hello_world**
2. **cd remoteproj:** navigate into your local working repository
3. **git branch –M main/git checkout –b main:** change/create your current branch name to main
4. **git remote add origin git@github.com:YourRepo.git:** create an alias origin for the remote repository URL
5. **git pull origin main:** update local repository to the newest commit
6. Make changes on local machine:
7. **git push –u origin main:** push the existing repository to Github

```
PS C:\Users\Chengtao\Desktop\GitDemo\remoteproj> git init
Initialized empty Git repository in C:/Users/Chengtao/Desktop/GitDemo/remoteproj/.git/
PS C:\Users\Chengtao\Desktop\GitDemo\remoteproj> git checkout -b main
Switched to a new branch 'main'
PS C:\Users\Chengtao\Desktop\GitDemo\remoteproj> git remote add origin git@github.com:ChengtaoJi/hello_world.git
PS C:\Users\Chengtao\Desktop\GitDemo\remoteproj> git pull origin main
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 8 (delta 0), reused 3 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (8/8), 3.37 KiB | 72.00 KiB/s, done.
From github.com:ChengtaoJi/hello_world
 * branch            main       -> FETCH_HEAD
 * [new branch]      main       -> origin/main
PS C:\Users\Chengtao\Desktop\GitDemo\remoteproj> ls


    Directory: C:\Users\Chengtao\Desktop\GitDemo\remoteproj


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----          3/9/2025   3:11 PM              2 READ.ME


PS C:\Users\Chengtao\Desktop\GitDemo\remoteproj> git add --all
PS C:\Users\Chengtao\Desktop\GitDemo\remoteproj> git commit -m "add a new file"
[main 14c61d3] add a new file
 1 file changed, 200 insertions(+)
 create mode 100644 index.html
PS C:\Users\Chengtao\Desktop\GitDemo\remoteproj> git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 20 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 1.79 KiB | 1.79 MiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:ChengtaoJi/hello_world.git
   88e11b3..14c61d3  main -> main
```


Xi'an Jiaotong-Liverpool University
西交利物浦大學

# *Git(Hub) Tutorial*

Chengtao Ji

(Materials from Online)

2025/3/10