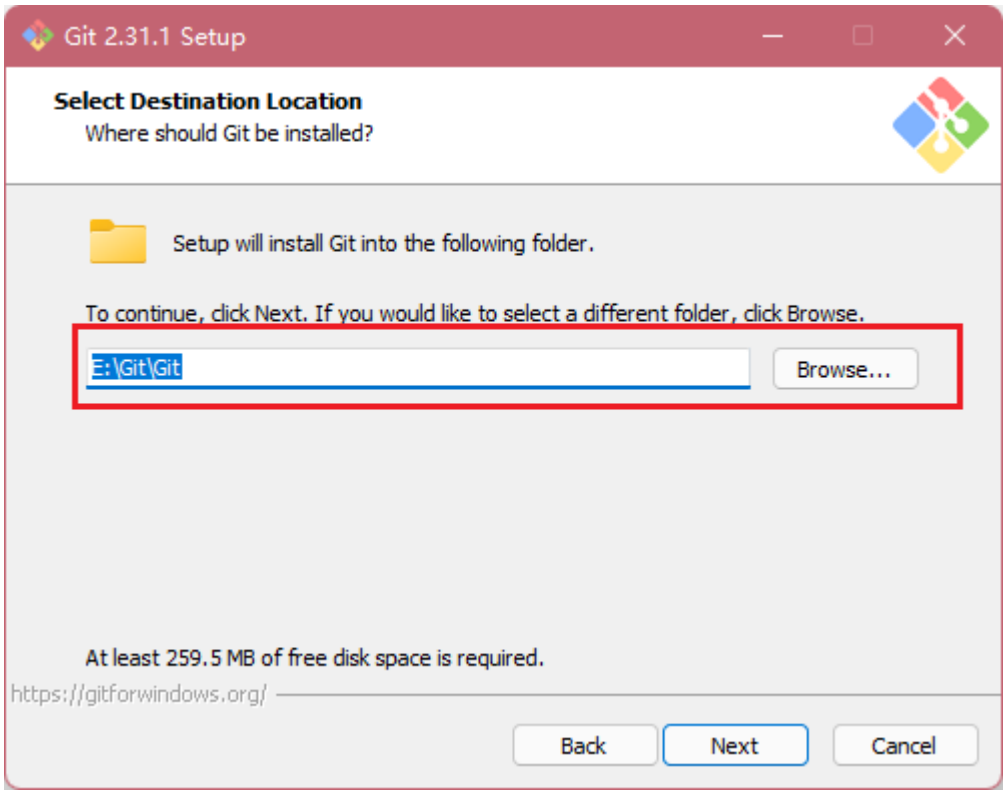
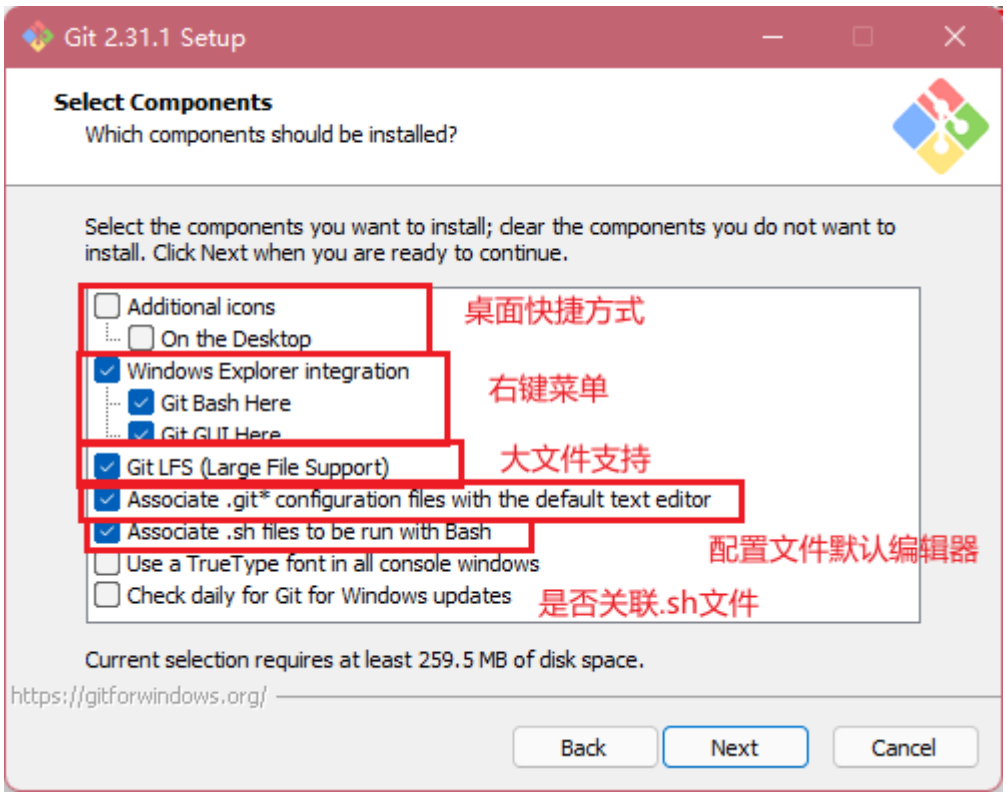
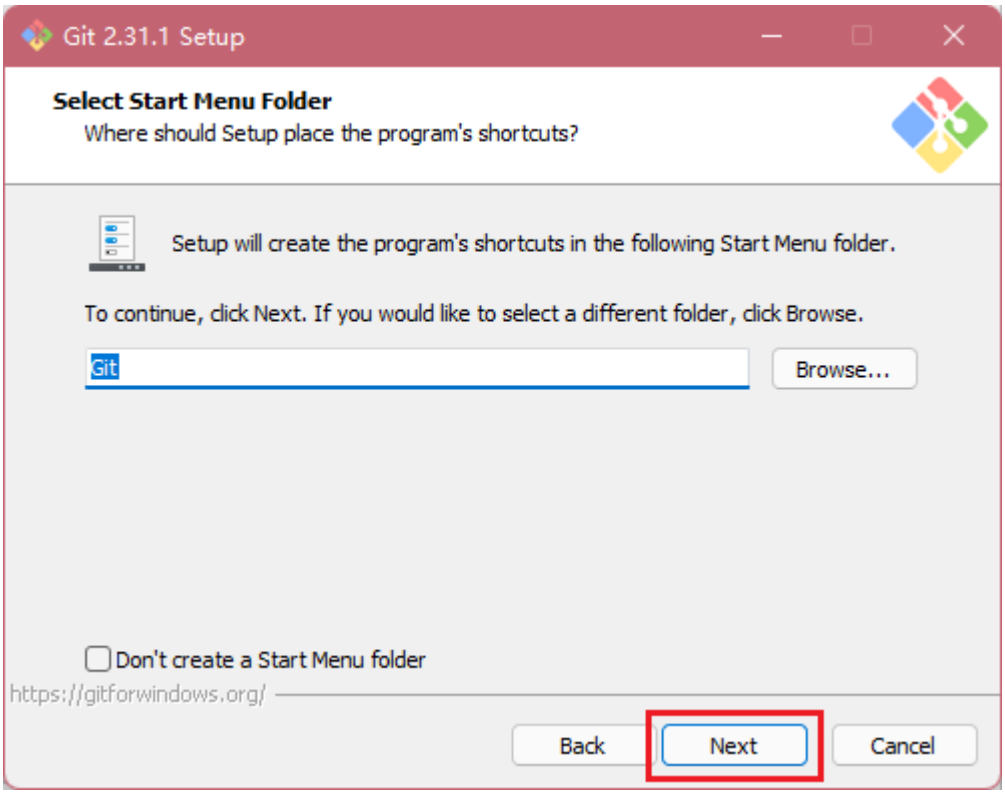
Git

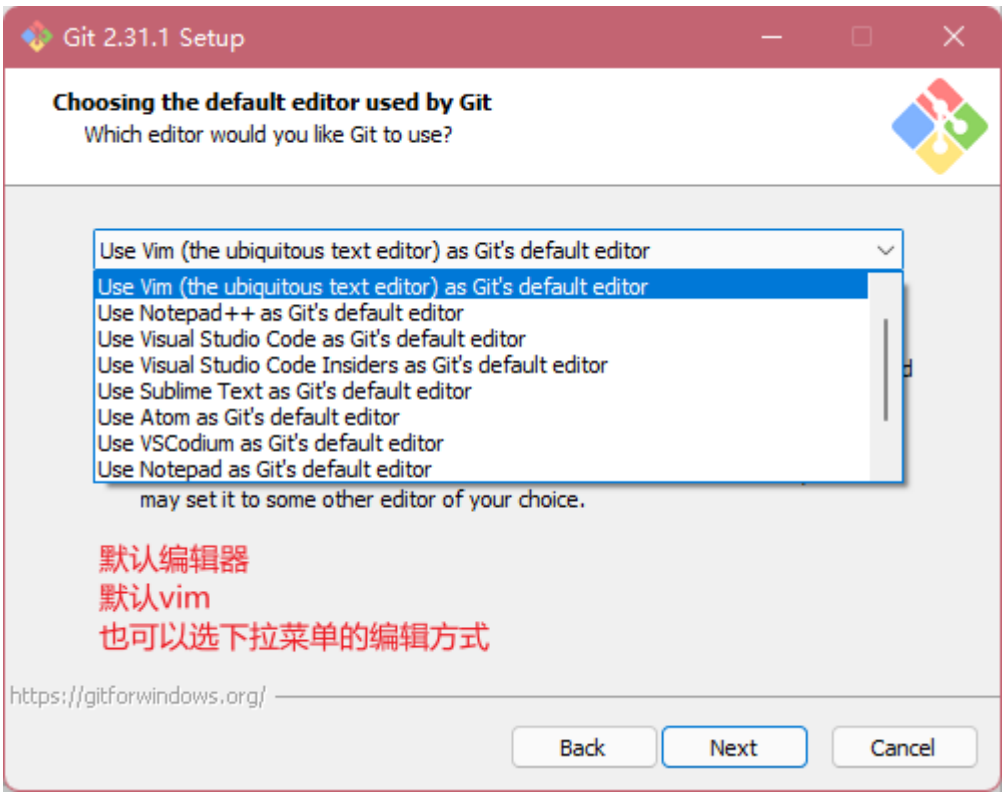
一、安装

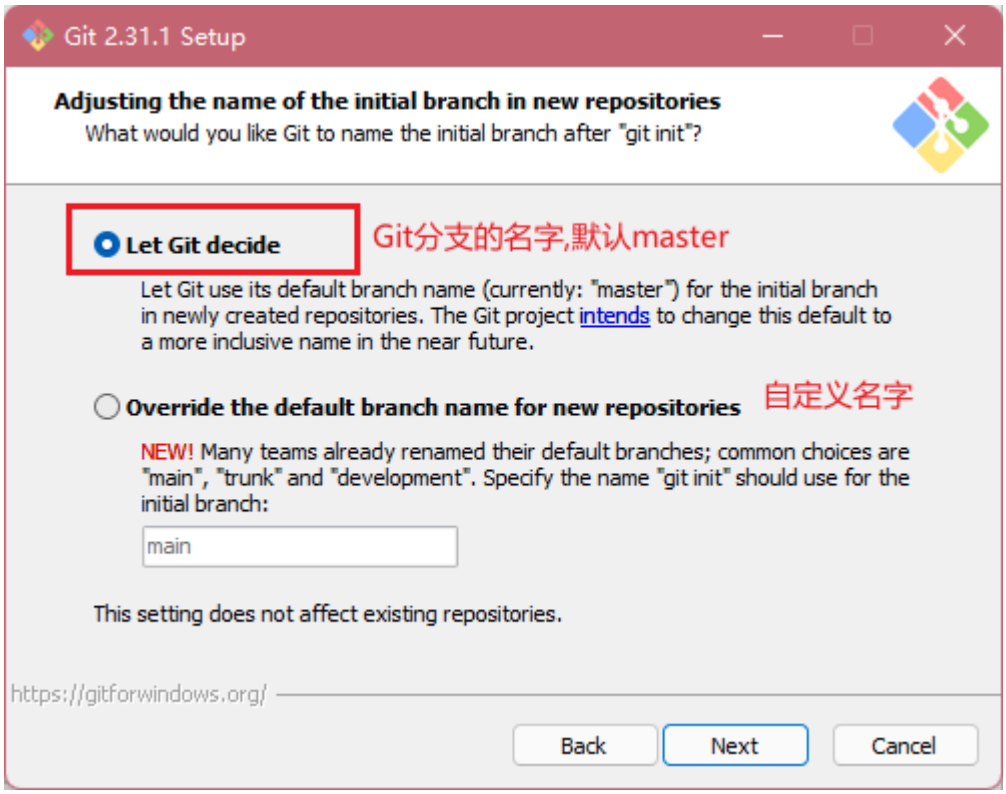
安装地址: https://npm.taobao.org/mirrors/git-for-windows/


Git 2.31.1 Setup
Select Destination Location
Where should Git be installed?
Setup will install Git into the following folder.
To continue, click Next. If you would like to select a different folder, click Browse.
E:\Git\Git
Browse...
At least 259.5 MB of free disk space is required.
https://gitforwindows.org/
Back
Next
Cancel

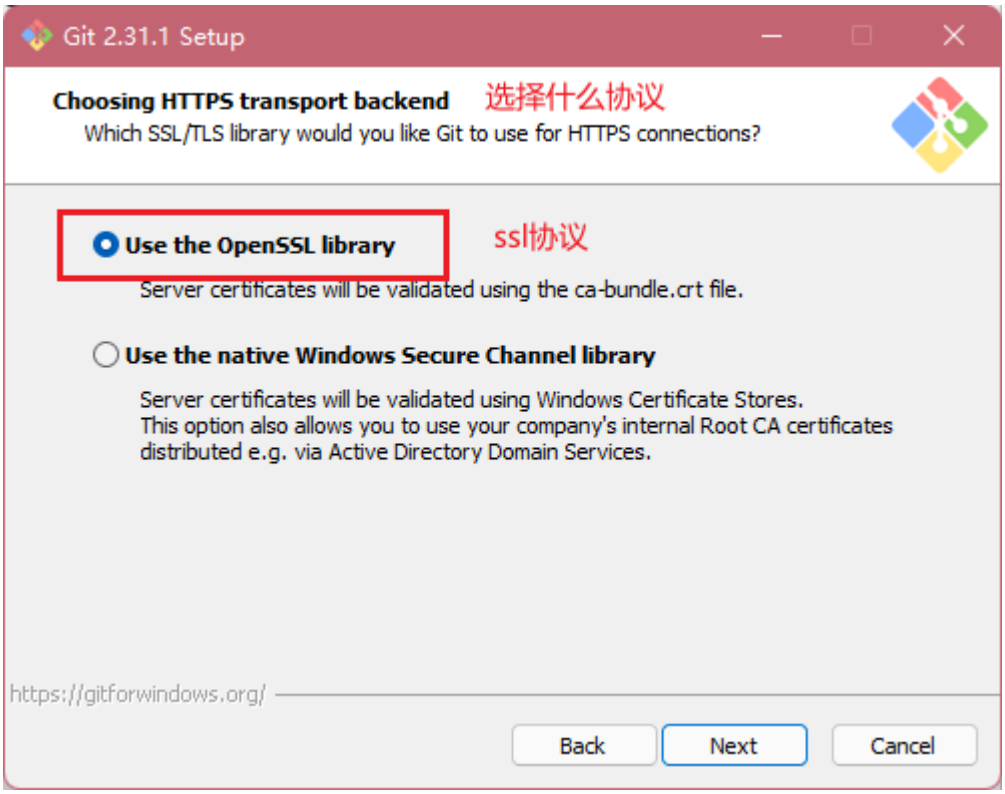

Git 2.31.1 Setup
Select Components
Which components should be installed?
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.
Additional icons
On the Desktop
桌面快捷方式
Windows Explorer integration
Git Bash Here
Git GUI Here
右键菜单
Git LFS (Large File Support)
大文件支持
Associate .git* configuration files with the default text editor
Associate .sh files to be run with Bash
配置文件默认编辑器
Use a TrueType font in all console windows
Check daily for Git for Windows updates
是否关联.sh文件
Current selection requires at least 259.5 MB of disk space.
https://gitforwindows.org/
Back
Next
Cancel

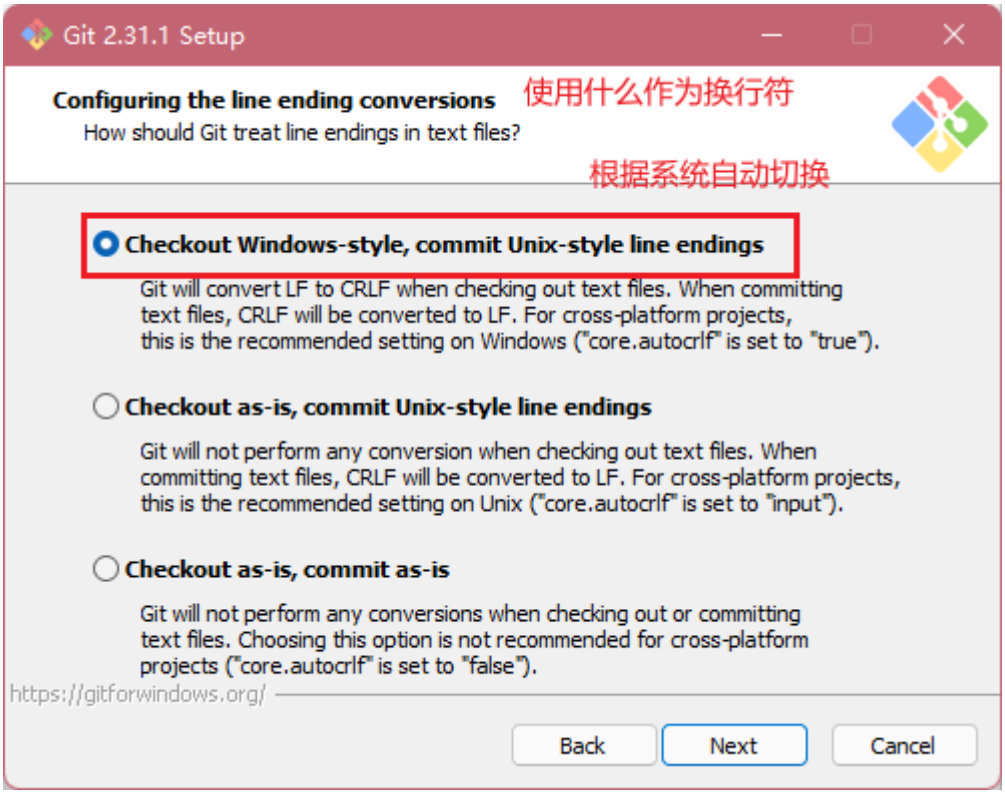

Git 2.31.1 Setup
Select Start Menu Folder
Where should Setup place the program's shortcuts?
Setup will create the program's shortcuts in the following Start Menu folder.
To continue, click Next. If you would like to select a different folder, click Browse.
Git
Browse...
Don't create a Start Menu folder
https://gitforwindows.org/
Back
Next
Cancel

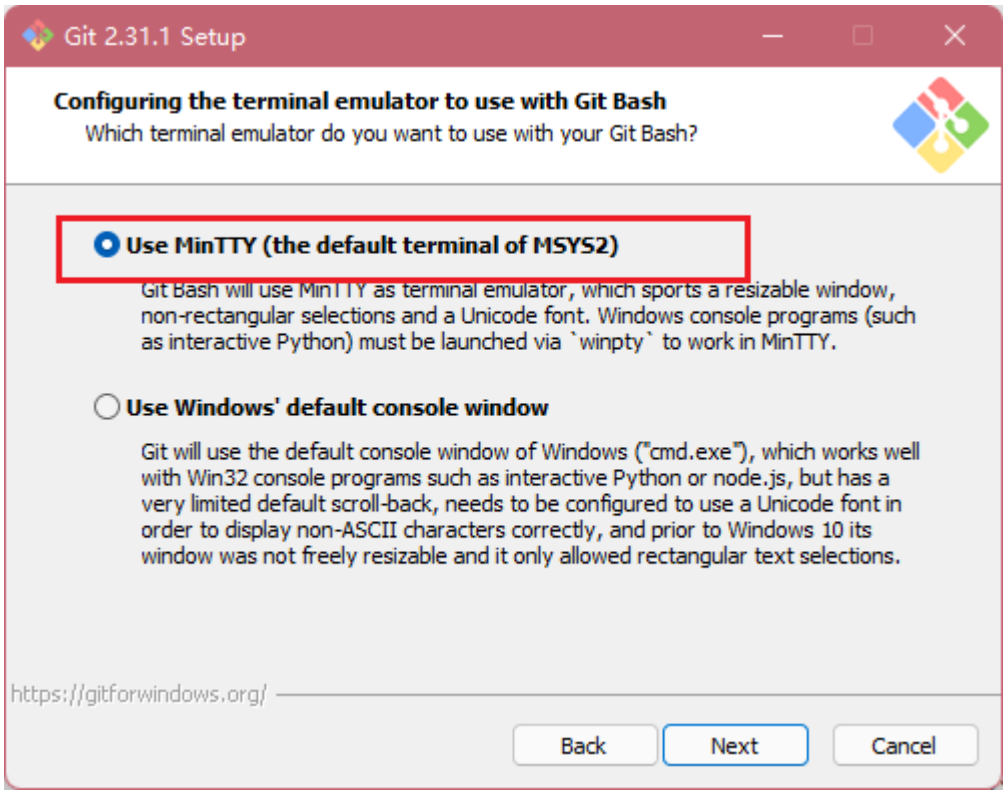

Git 2.31.1 Setup
Choosing the default editor used by Git
Which editor would you like Git to use?
Use Vim (the ubiquitous text editor) as Git's default editor
Use Vim (the ubiquitous text editor) as Git's default editor
Use Notepad++ as Git's default editor
Use Visual Studio Code as Git's default editor
Use Visual Studio Code Insiders as Git's default editor
Use Sublime Text as Git's default editor
Use Atom as Git's default editor
Use VSCodium as Git's default editor
Use Notepad as Git's default editor
may set it to some other editor of your choice.
默认编辑器
默认vim
也可以选下拉菜单的编辑方式
https://gitforwindows.org/
Back
Next
Cancel

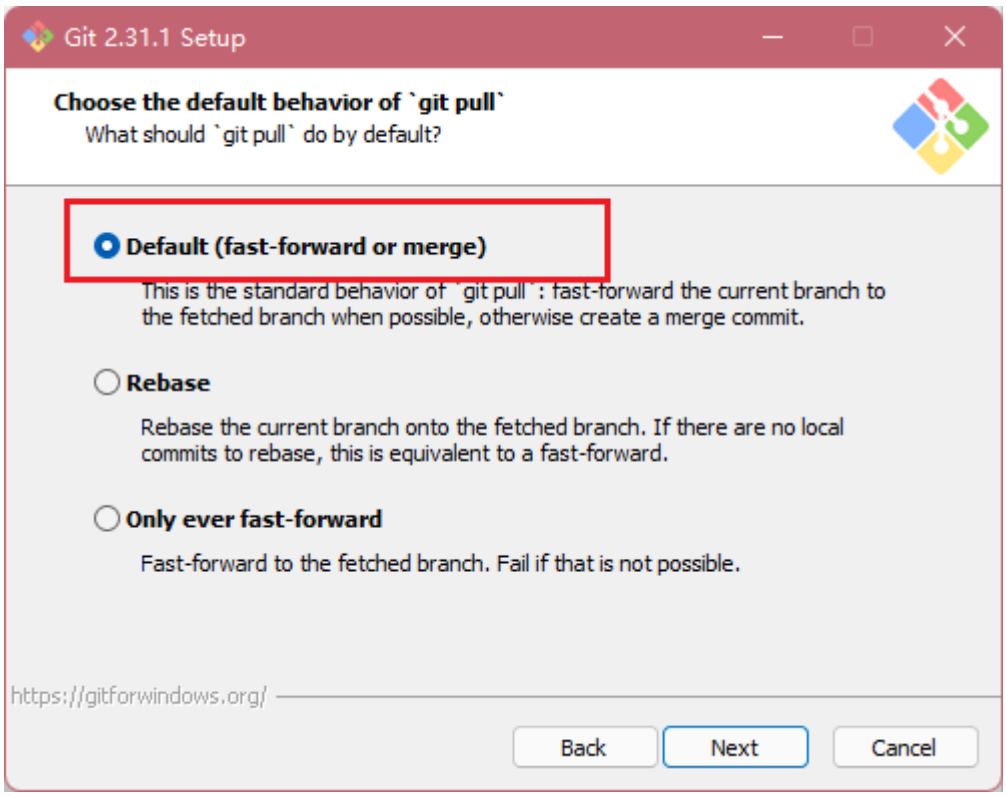

Git 2.31.1 Setup
Adjusting the name of the initial branch in new repositories
What would you like Git to name the initial branch after "git init"?
Let Git decide
Git分支的名字,默认master
Let Git use its default branch name (currently: "master") for the initial branch in newly created repositories. The Git project intends to change this default to a more inclusive name in the near future.
Override the default branch name for new repositories
自定义名字
NEW! Many teams already renamed their default branches; common choices are "main", "trunk" and "development". Specify the name "git init" should use for the initial branch:
main
This setting does not affect existing repositories.
https://gitforwindows.org/
Back
Next
Cancel

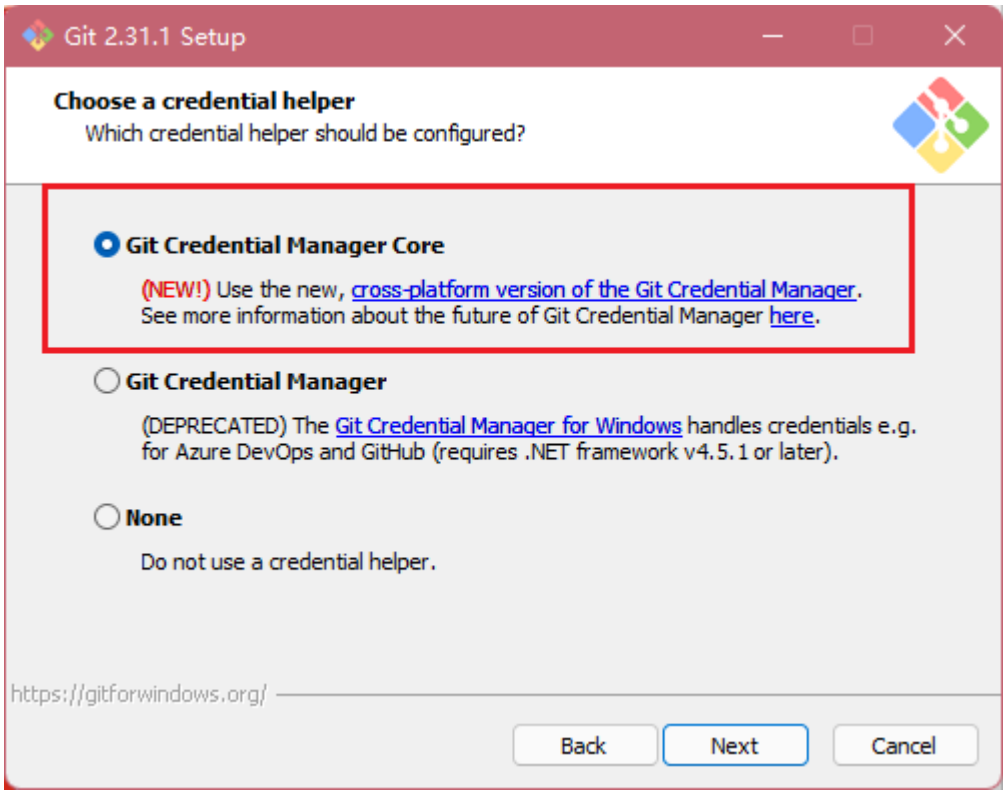

Git 2.31.1 Setup
Adjusting your PATH environment
How would you like to use Git from the command line?
Use Git from Git Bash only
只在Git默认控制台能使用
This is the most cautious choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.
Git from the command line and also from 3rd-party software
可在cmd中使用git
(Recommended) This option adds only some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools.
You will be able to use Git from Git Bash, the Command Prompt and the Windows PowerShell as well as any third-party software looking for Git in PATH.
Use Git and optional Unix tools from the Command Prompt
狗都不用
Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.
https://gitforwindows.org/
Back
Next
Cancel

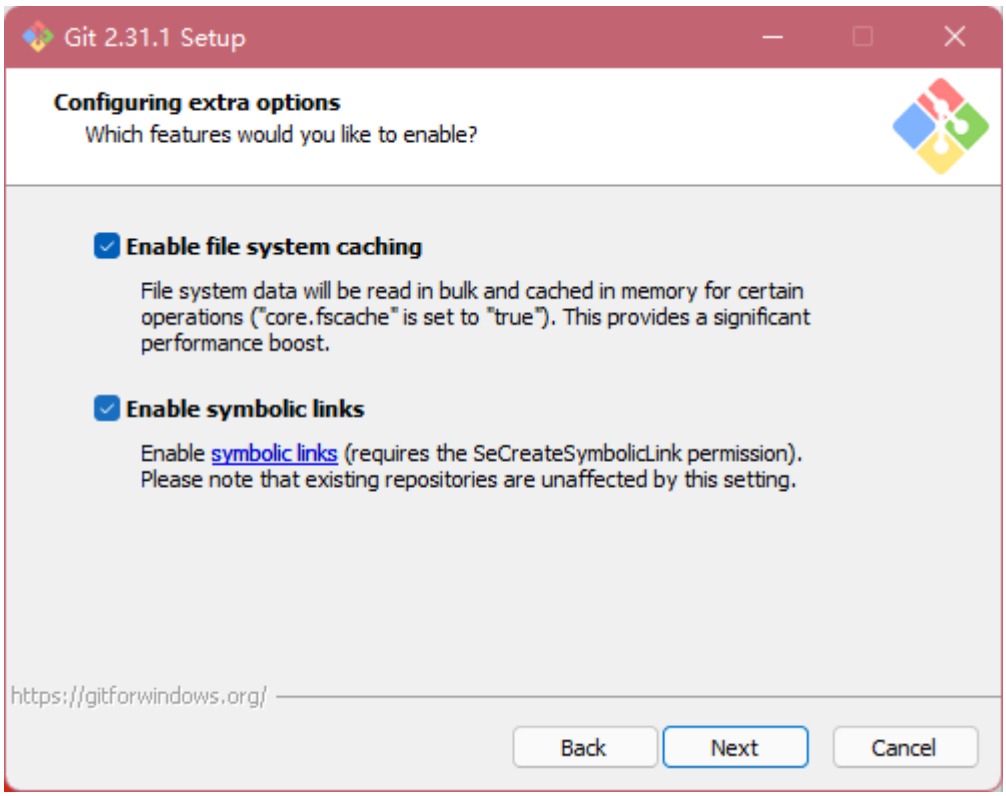

Git 2.31.1 Setup
Choosing HTTPS transport backend
选择什么协议
Which SSL/TLS library would you like Git to use for HTTPS connections?
Use the OpenSSL library
ssl协议
Server certificates will be validated using the ca-bundle.crt file.
Use the native Windows Secure Channel library
Server certificates will be validated using Windows Certificate Stores.
This option also allows you to use your company's internal Root CA certificates
distributed e.g. via Active Directory Domain Services.
https://gitforwindows.org/
Back
Next
Cancel

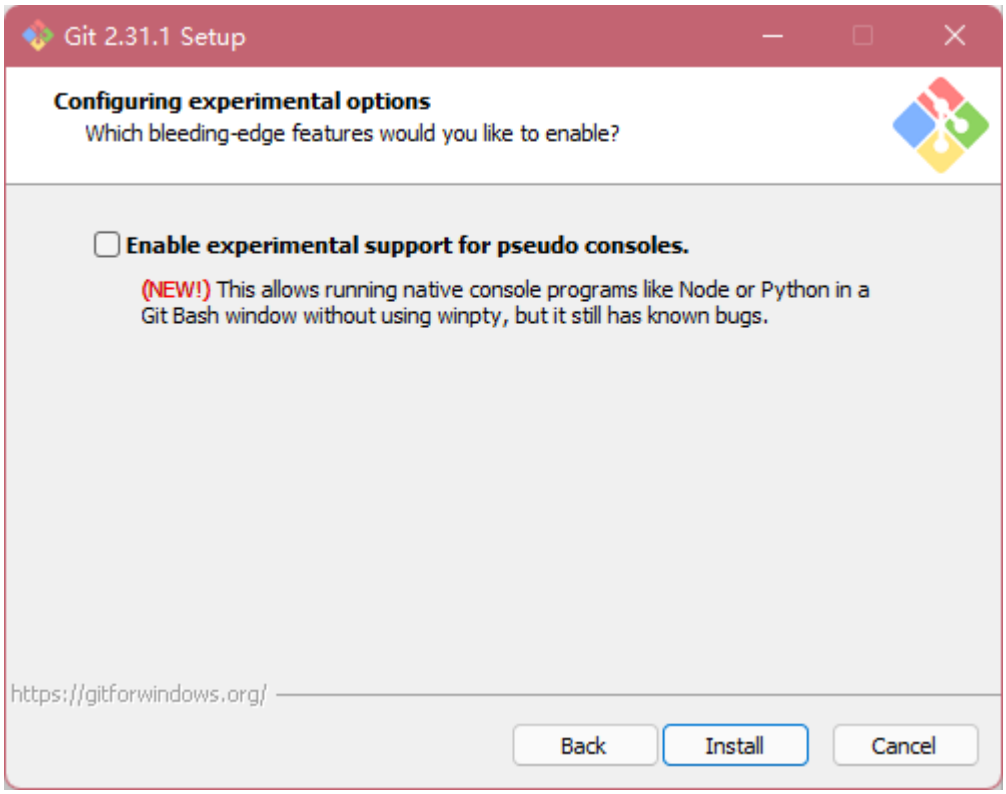

Git 2.31.1 Setup
Configuring the line ending conversions
使用什么作为换行符
How should Git treat line endings in text files?
根据系统自动切换
Checkout Windows-style, commit Unix-style line endings
Git will convert LF to CRLF when checking out text files. When committing
text files, CRLF will be converted to LF. For cross-platform projects,
this is the recommended setting on Windows ("core.autocrlf" is set to "true").
Checkout as-is, commit Unix-style line endings
Git will not perform any conversion when checking out text files. When
committing text files, CRLF will be converted to LF. For cross-platform projects,
this is the recommended setting on Unix ("core.autocrlf" is set to "input").
Checkout as-is, commit as-is
Git will not perform any conversions when checking out or committing
text files. Choosing this option is not recommended for cross-platform
projects ("core.autocrlf" is set to "false").
https://gitforwindows.org/
Back
Next
Cancel

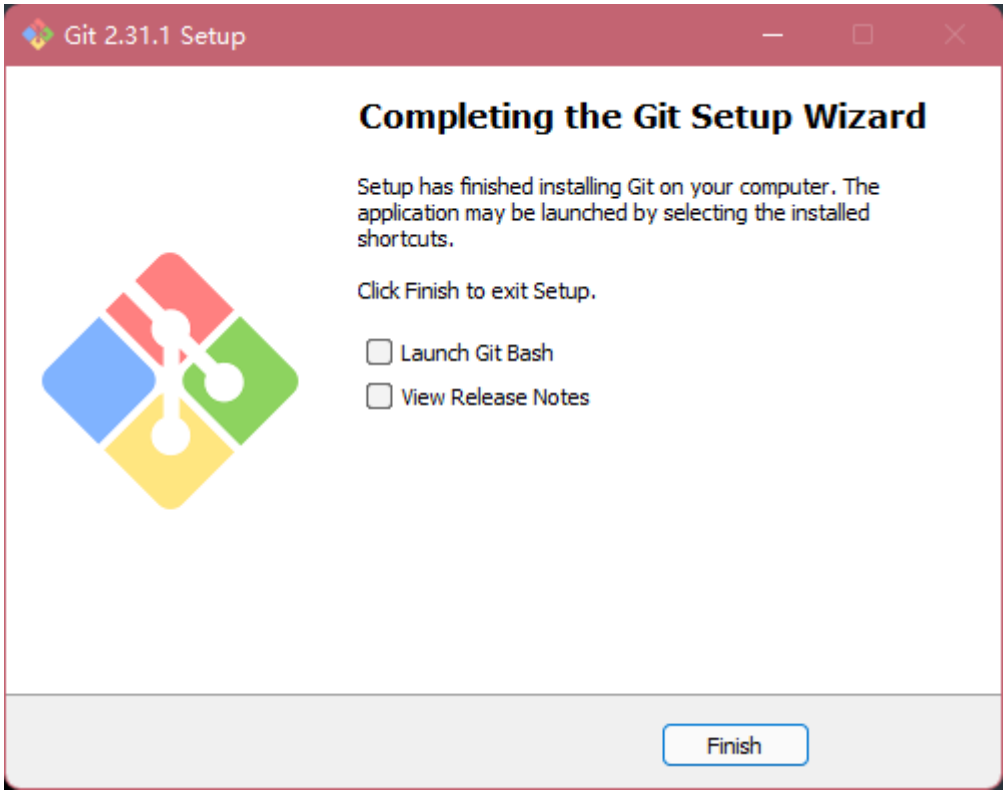

Git 2.31.1 Setup
Configuring the terminal emulator to use with Git Bash
Which terminal emulator do you want to use with your Git Bash?
Use MinTTY (the default terminal of MSYS2)
Git Bash will use MinTTY as terminal emulator, which sports a resizable window, non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via `winpty` to work in MinTTY.
Use Windows' default console window
Git will use the default console window of Windows ("cmd.exe"), which works well with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.
https://gitforwindows.org/
Back
Next
Cancel


Git 2.31.1 Setup
Choose the default behavior of `git pull`
What should `git pull` do by default?
Default (fast-forward or merge)
This is the standard behavior of `git pull`: fast-forward the current branch to the fetched branch when possible, otherwise create a merge commit.
Rebase
Rebase the current branch onto the fetched branch. If there are no local commits to rebase, this is equivalent to a fast-forward.
Only ever fast-forward
Fast-forward to the fetched branch. Fail if that is not possible.
https://gitforwindows.org/
Back
Next
Cancel

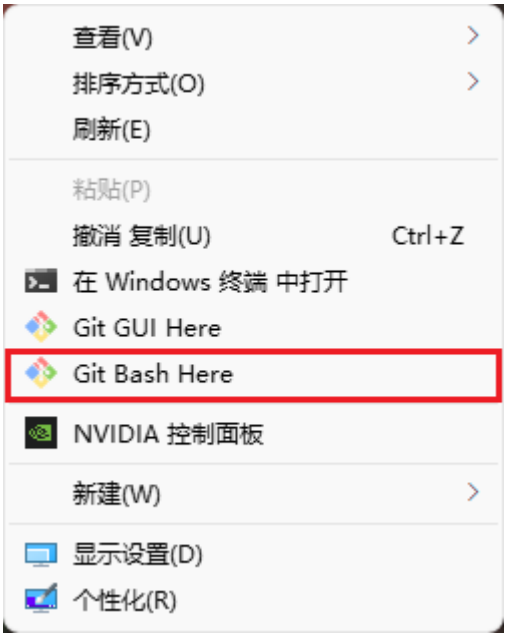

Git 2.31.1 Setup
Choose a credential helper
Which credential helper should be configured?
Git Credential Manager Core
(NEW!) Use the new, cross-platform version of the Git Credential Manager.
See more information about the future of Git Credential Manager here.
Git Credential Manager
(DEPRECATED) The Git Credential Manager for Windows handles credentials e.g.
for Azure DevOps and GitHub (requires .NET framework v4.5.1 or later).
None
Do not use a credential helper.
https://gitforwindows.org/
Back
Next
Cancel

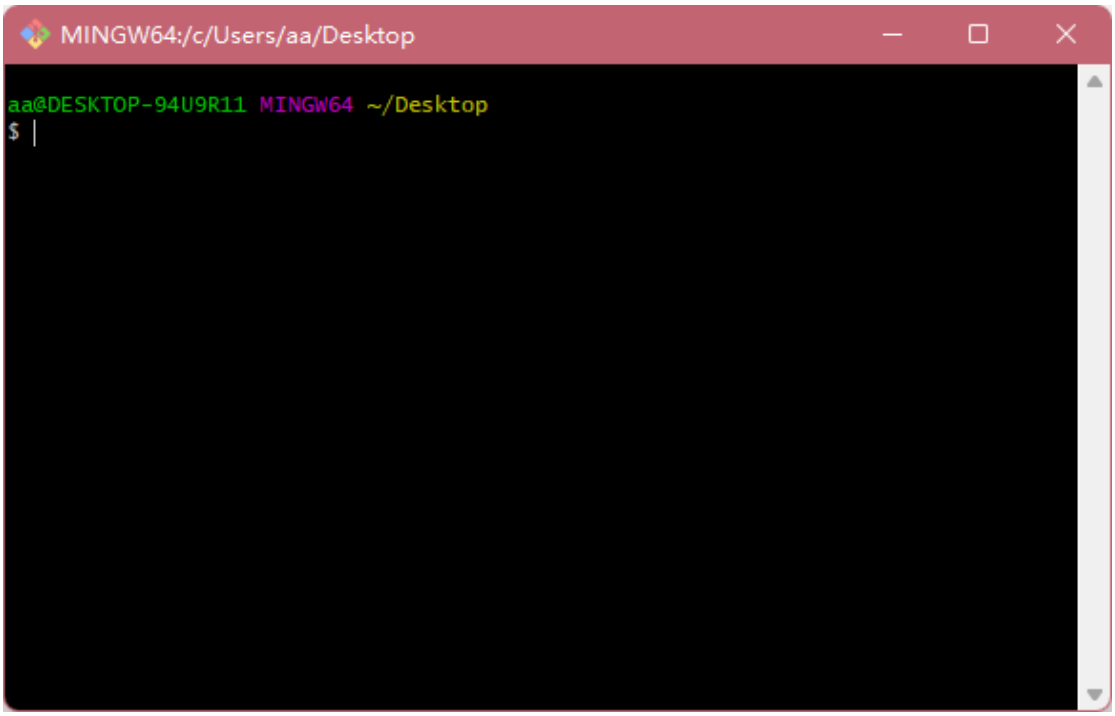

Git 2.31.1 Setup
Configuring extra options
Which features would you like to enable?
Enable file system caching
File system data will be read in bulk and cached in memory for certain
operations ("core.fscache" is set to "true"). This provides a significant
performance boost.
Enable symbolic links
Enable symbolic links (requires the SeCreateSymbolicLink permission).
Please note that existing repositories are unaffected by this setting.
https://gitforwindows.org/
Back
Next
Cancel


Git 2.31.1 Setup
Configuring experimental options
Which bleeding-edge features would you like to enable?
Enable experimental support for pseudo consoles.
(NEW!) This allows running native console programs like Node or Python in a Git Bash window without using winpty, but it still has known bugs.
https://gitforwindows.org/
Back
Install
Cancel


Git 2.31.1 Setup
Completing the Git Setup Wizard
Setup has finished installing Git on your computer. The application may be launched by selecting the installed shortcuts.
Click Finish to exit Setup.
Launch Git Bash
View Release Notes
Finish

打开





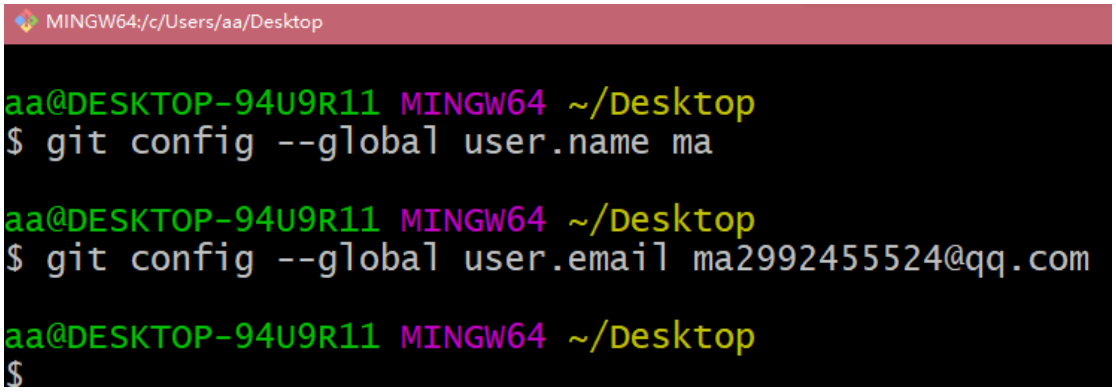
二、Git 常用命令

命令名称	作用
git config --global user.name 用户名	设置用户签名
git config --global user.email	设置用户签名

邮箱	
git init	初始化本地库
git status	查看本地库状态
git add 文件名	添加到暂存区
git commit -m "日志信息" 文件名	提交到本地库
git reflog	查看历史记录
git reset --hard 版本号	版本穿梭

(一)、创建用户签名

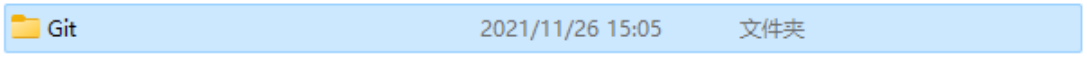
签名的作用是区分不同操作者身份。用户的签名信息在每一个版本的提交信息中能够看 到，以此确认本次提交是谁做的。Git 首次安装必须设置一下用户签名，否则无法提交代码。



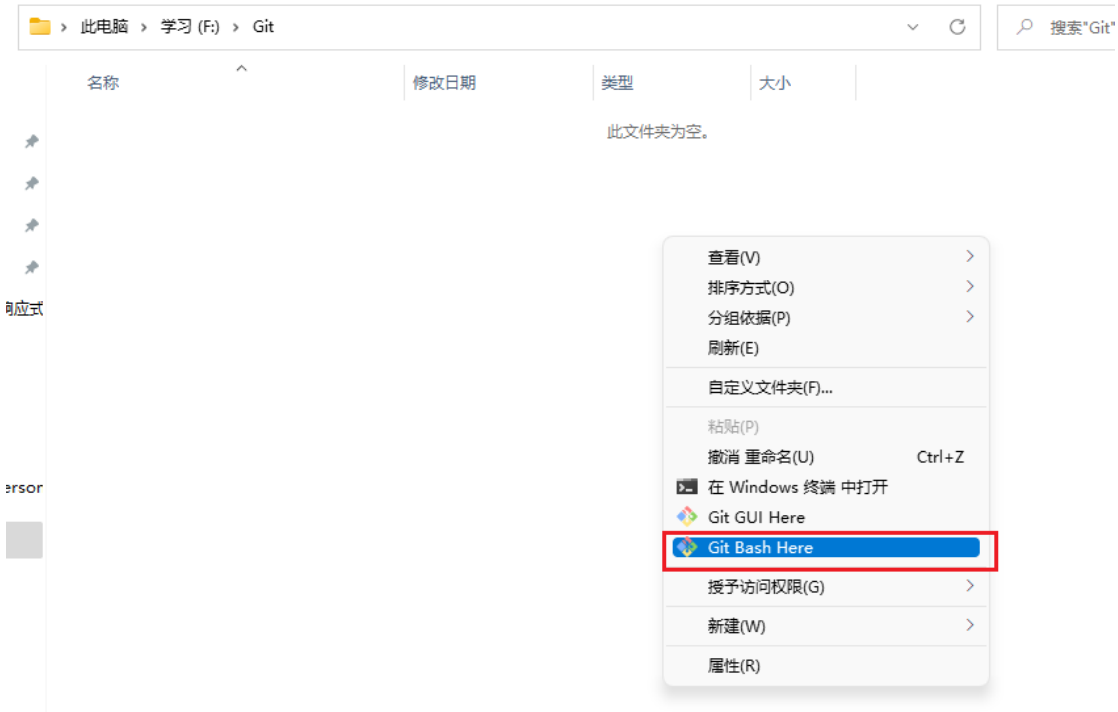
在 C:\Users\aa\.gitconfig 中查看

(二)、初始化本地库

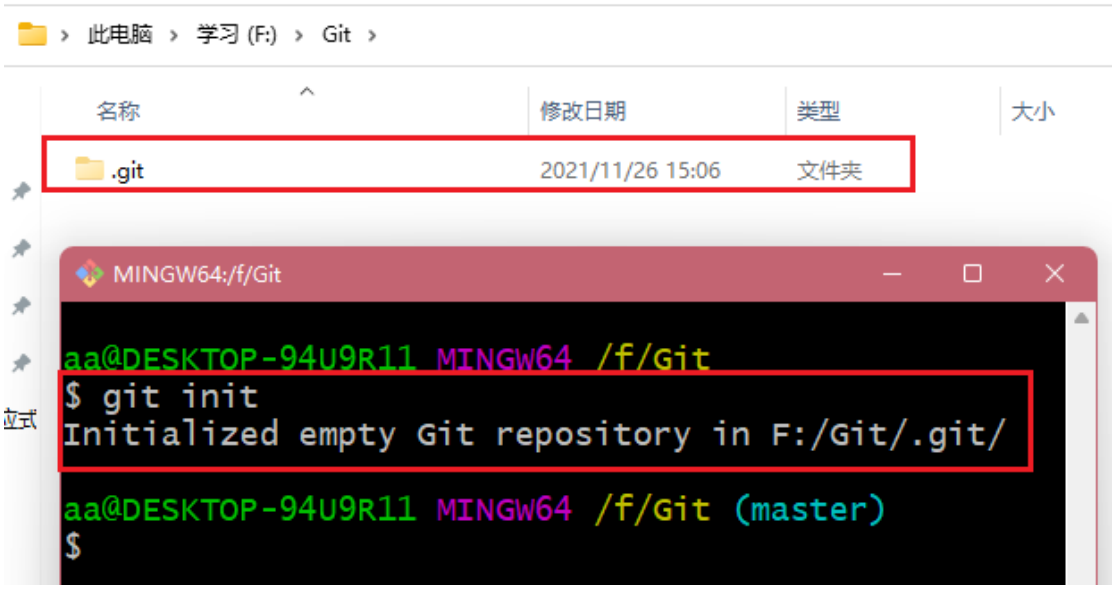
1、进入一个项目中



2、右键打开 git bashHere

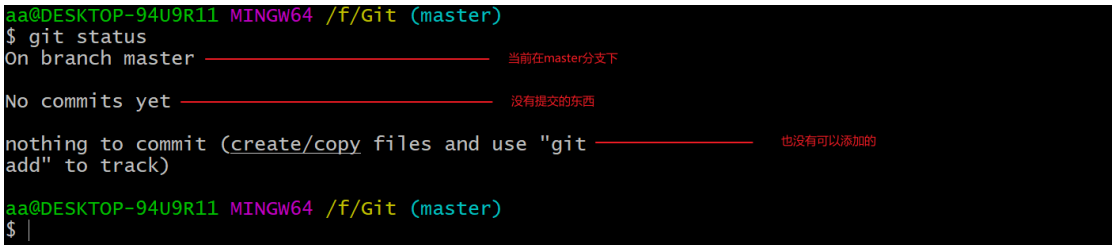


3、输入 git init 会生成一个.git 的文件

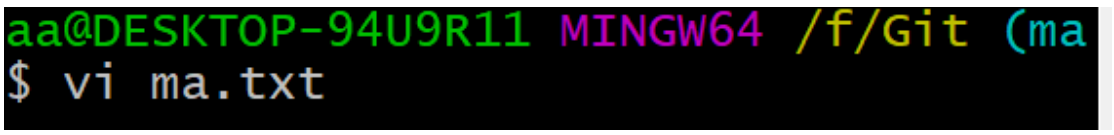


(三)、查看 git 状态

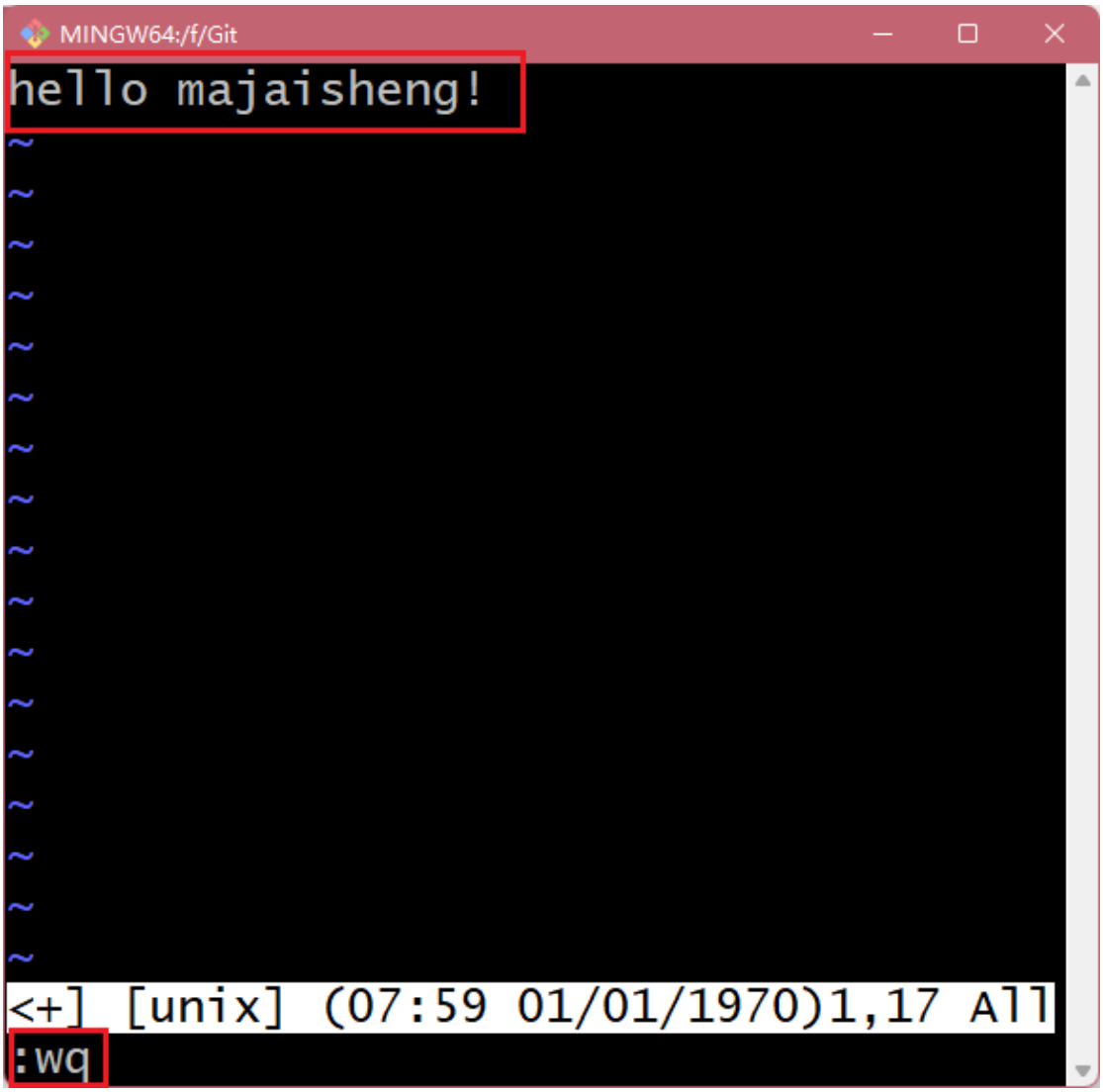
1、Git　status



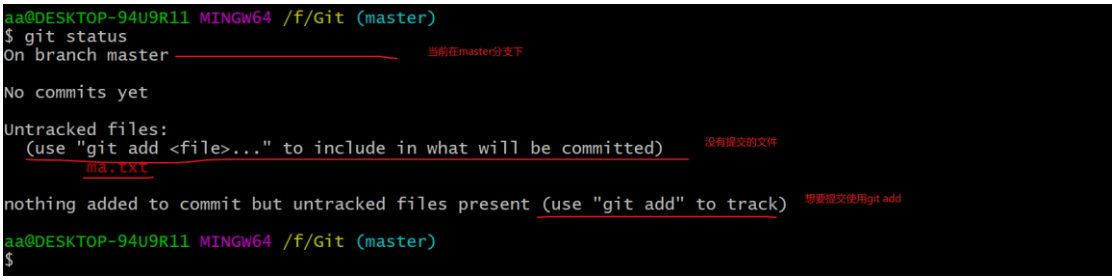
2、创建一个文件 vi ma.txt



3、输入内容保存退出

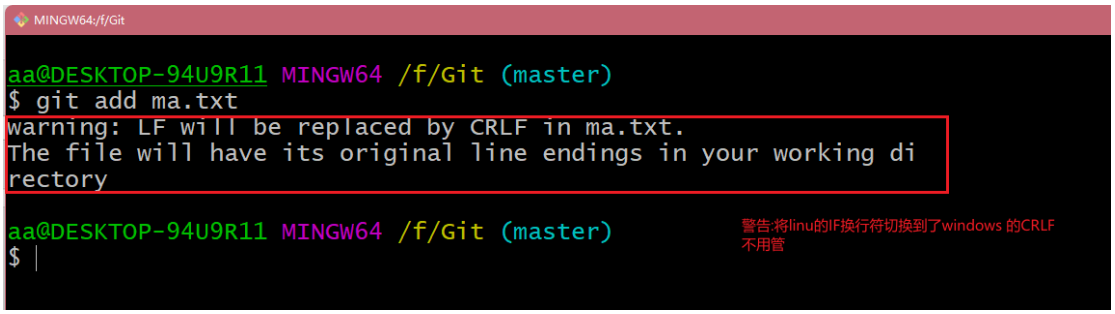


4、再次使用 git status 命令



(四)、添加删除暂存区

1、git add 文件名

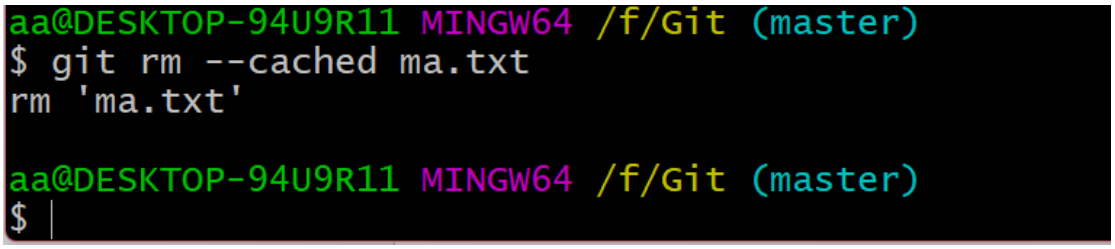


2、查看状态

Git status

3、删除暂存区的文件

Git rm - -cached 文件名



4、再次查看状态

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git status
On branch master

No commits yet

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

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

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

说明删除只是删除了暂存区的文件,而本地的文件没有删除

5、再次提交

Git add 文件名

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git add ma.txt
warning: LF will be replaced by CRLF in ma.txt.
The file will have its original line endings in your working directory

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

(五)、提交到远程仓库

1、git commit –m “my one commit” 文件名

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git commit -m "my one commit" ma.txt
warning: LF will be replaced by CRLF in ma.txt.
The file will have its original line endings in your working directory
[master (root-commit) e33763f] my one commit
 1 file changed, 1 insertion(+)
 create mode 100644 ma.txt

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

切换回车

master分支 e33763f版本号
1个文件 1行插入
创建模块 内存100644 文件名

2、查看状态

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git status
On branch master
nothing to commit, working tree clean

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

master分支下
工作区是空的

3、git reflog 查看简单信息

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git reflog
e33763f (HEAD -> master) HEAD@{0}: commit (initial): my one commit
```

4、git log 查看详细信息

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git log
commit e33763fb1535369437325367ff34ecefd178ce6f (HEAD -> master)
Author: ma <ma2992455524@qq.com>
Date:   Fri Nov 26 15:30:25 2021 +0800

    my one commit

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

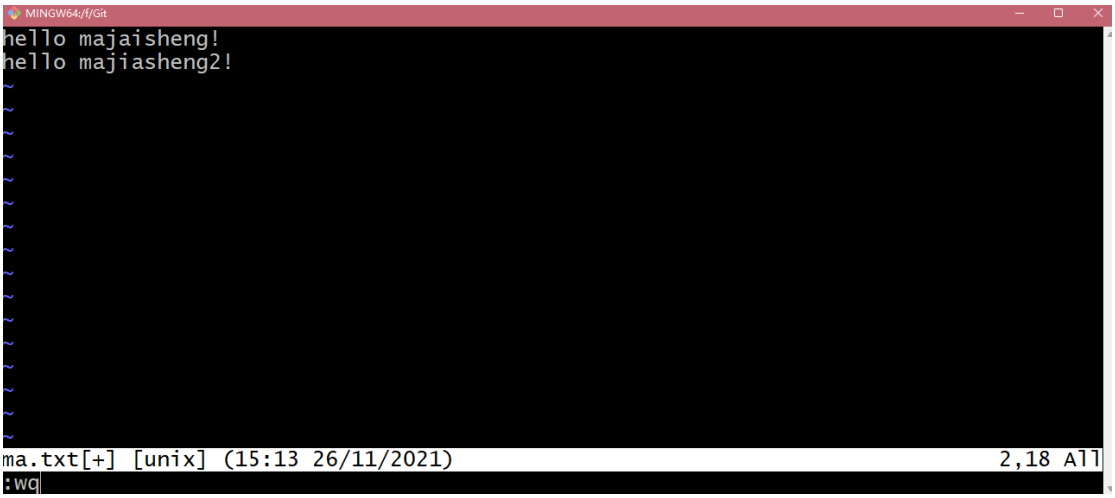
完整版本号
用户签名
什么时候提交的
提交时候的说明

(六)、修改文件

1、Vi ma.txt

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ vi ma.txt
```

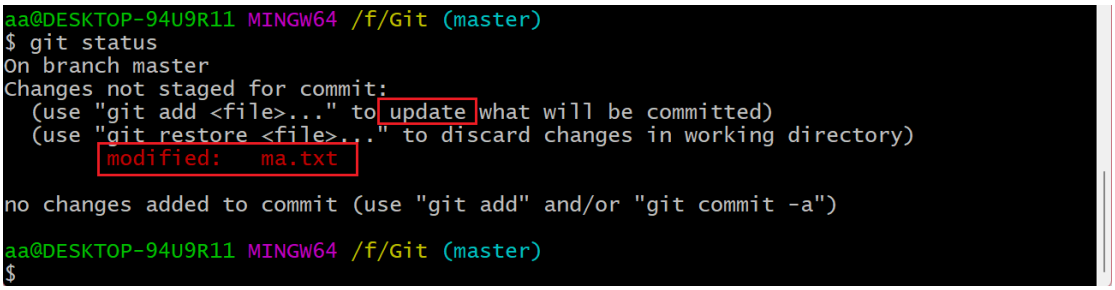
2、添加一列



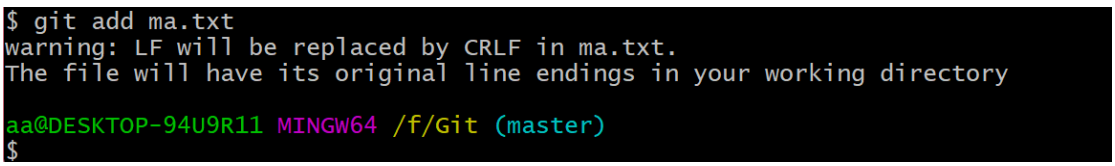
3、查看状态

Git status

显示当前有数据更新,并且没被 git 追踪到



4、使用 git add 文件名上传到缓存区



5、使用 git commit –m “my two commit” 文件名

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git commit -m "my two commit"
[master 8b91922] my two commit
 1 file changed, 1 insertion(+)
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

一行添加

6、查看版本

Git reflog

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git reflog
8b91922 (HEAD -> master) HEAD@{0}: commit: my two commit
e33763f HEAD@{1}: commit (initial): my one commit

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

(七)、版本穿梭

1、首先查看 git 中有多少个版本

Git reflog

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git reflog
8b91922 (HEAD -> master) HEAD@{0}: commit: my two commit
e33763f HEAD@{1}: commit (initial): my one commit

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

2、查看当前版本的文件信息

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ cat ma.txt
hello majaisheng!
hello majiasheng2!

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

3、git refset –hard 版本号 穿梭到第一个版本

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git reset --hard e33763f
HEAD is now at e33763f my one commit
```

4、查看文件信息

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ cat ma.txt
hello majaisheng!

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

5、穿梭到最新版本

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git reset --hard 8b91922
HEAD is now at 8b91922 my two commit

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

6、查看文件

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ cat ma.txt
hello majaisheng!
hello majiasheng2!

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

(八)、分支

1、分支的创建

Git branch myFenZhi

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git branch myFenZhi
```

2、查看分支

Git branch -v

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git branch -v
* master   8b91922 my two commit
  myFenZhi 8b91922 my two commit

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

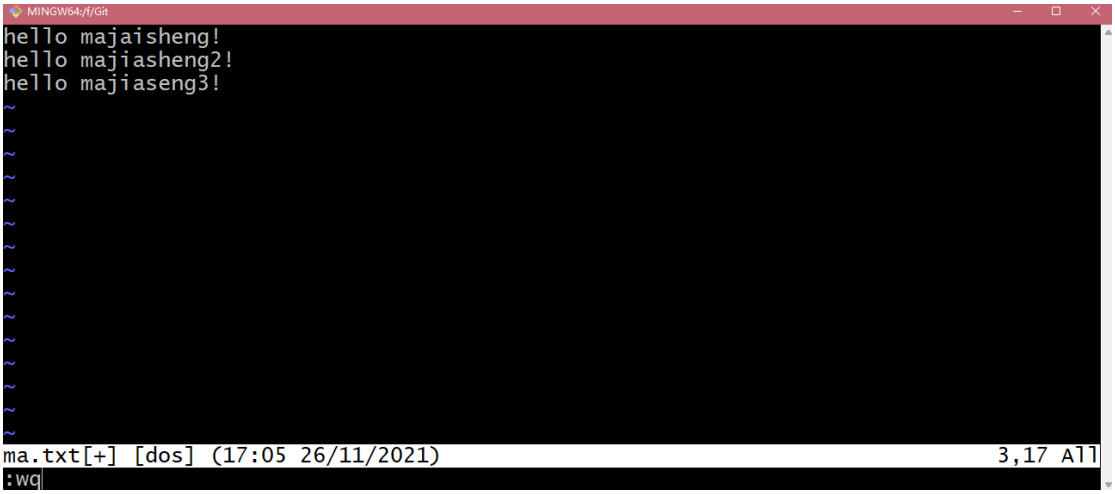
3、切换分支

Git checkout myFenZhi

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ Git checkout myFenZhi
Switched to branch 'myFenZhi'

aa@DESKTOP-94U9R11 MINGW64 /f/Git (myFenZhi)
$
```

4、在分支下修改文件

Vi ma.txt



5、切换到主分支

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (myFenZhi)
$ git checkout master
Switched to branch 'master'
M       ma.txt

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

6、查看文件

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (myFenZhi)
$ cat ma.txt
hello majaisheng!
hello majiasheng2!
```

(九)、合并分支(正常合并)

1、首先进入 master 主分支

Git merge master-1

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git merge master-1
Updating 4d2d777..888a403
Fast-forward
 ma.txt | 1 +
 1 file changed, 1 insertion(+)
```

2、查看文件

Cat ma.txt

就能看到 master-01 上传本地仓库中 ma.txt 中的 aaa 添加到了 mastet 的 ma.txt 中

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ cat ma.txt
hello majaisheng!
aaaaa
```

(十)、合并分支(冲突合并)

1、设置 master 的文件

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ cat ma.txt
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng! 2222
hello majaisheng!

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

2、设置 master-1 文件

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master-1)
$ cat ma.txt
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng! 333333

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master-1)
```

3、进入 master

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master-1)
$ git checkout master
Switched to branch 'master'
```

4、进行合并

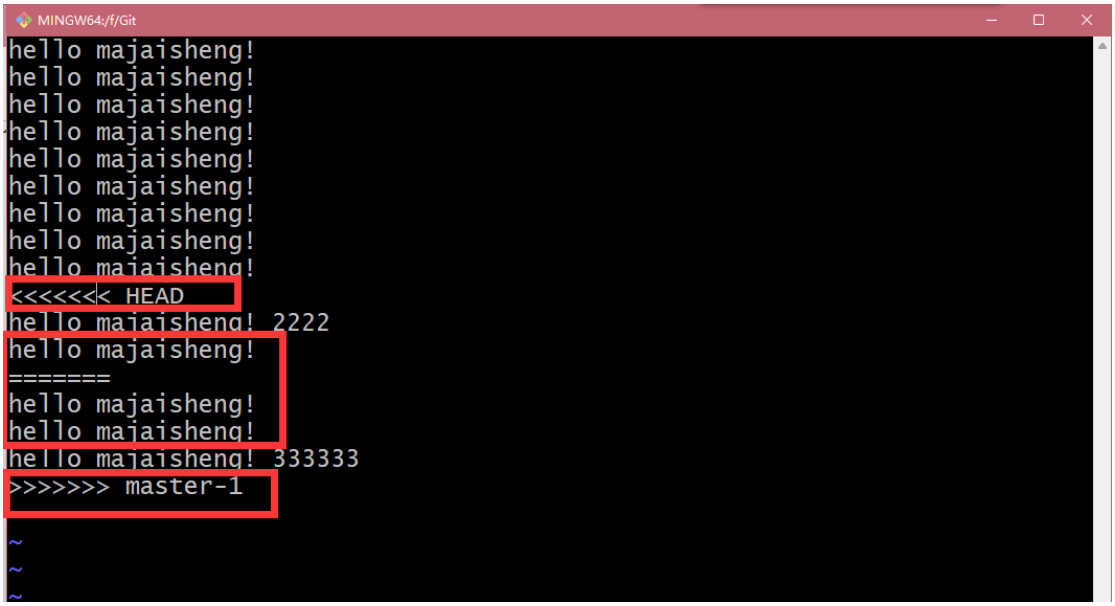
此时发现状态是合并中,不是真正的 master

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git merge master-1
Auto-merging ma.txt
CONFLICT (content): Merge conflict in ma.txt
Automatic merge failed; fix conflicts and then commit the result.

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master|MERGING)
$
```

5、进入 ma.txt 文件

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master|MERGING)
$ vi ma.txt
```

6、删除冲突的地方



7、添加到缓存区

8、添加到本地仓库不添加文件名

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master|MERGING)
$ git commit -m "chongtu hou"
[master 6515871] chongtu hou

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

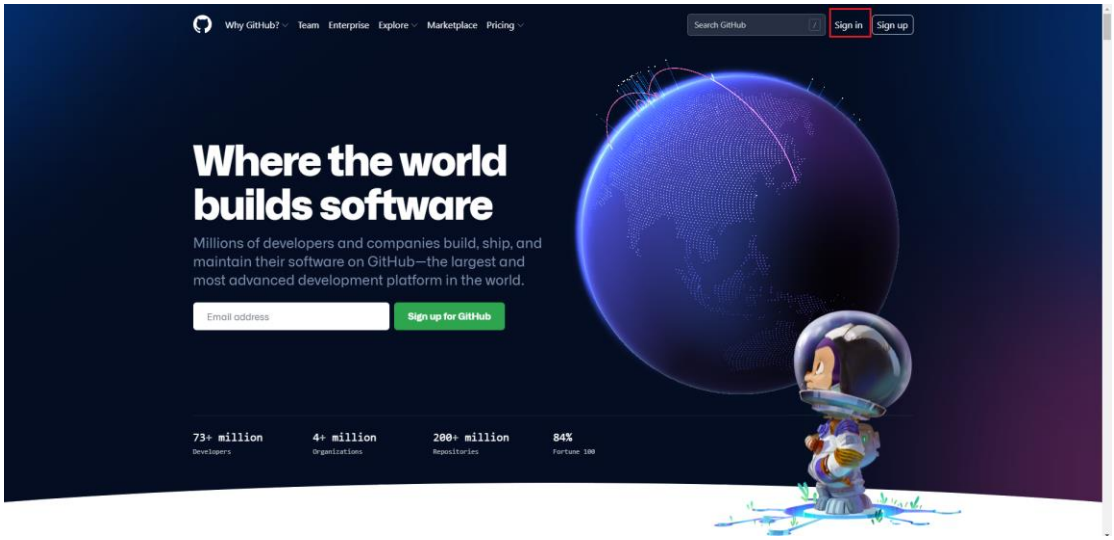
这时候就合并好了

三、GitHub

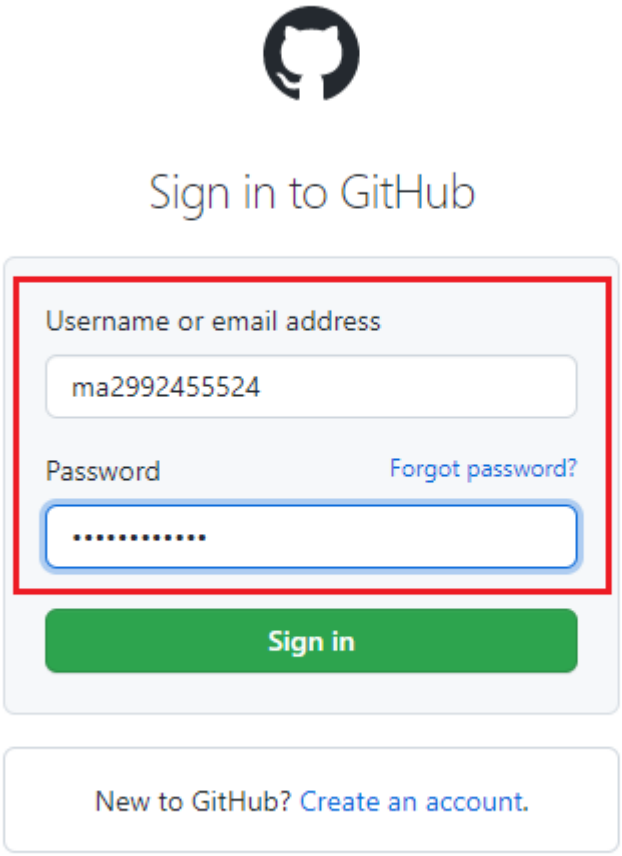
(一)、创建远程库&起别名

1、登录 github

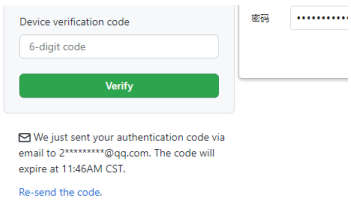
地址:https://github.com/

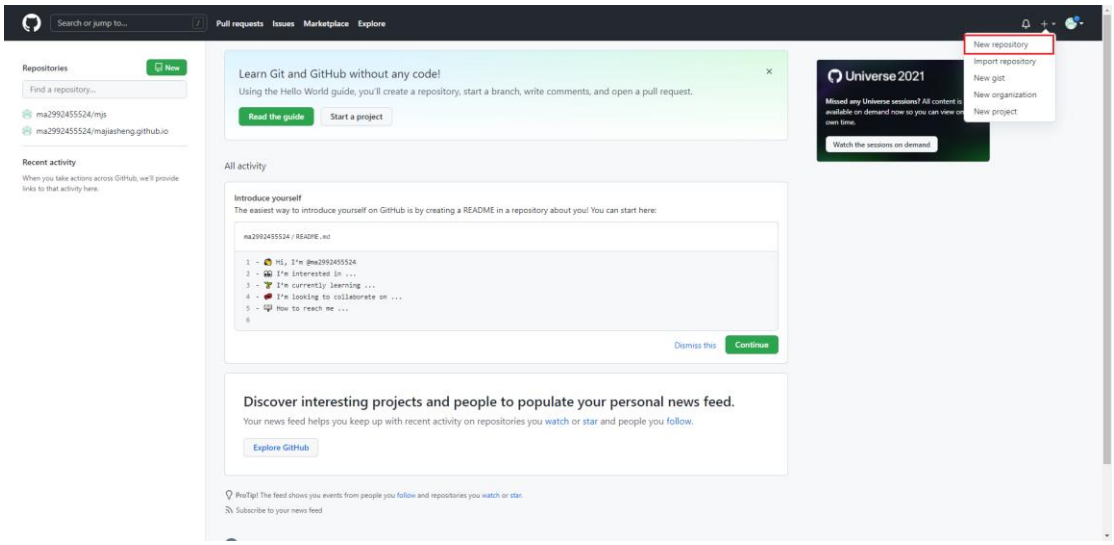


2、输入账号密码

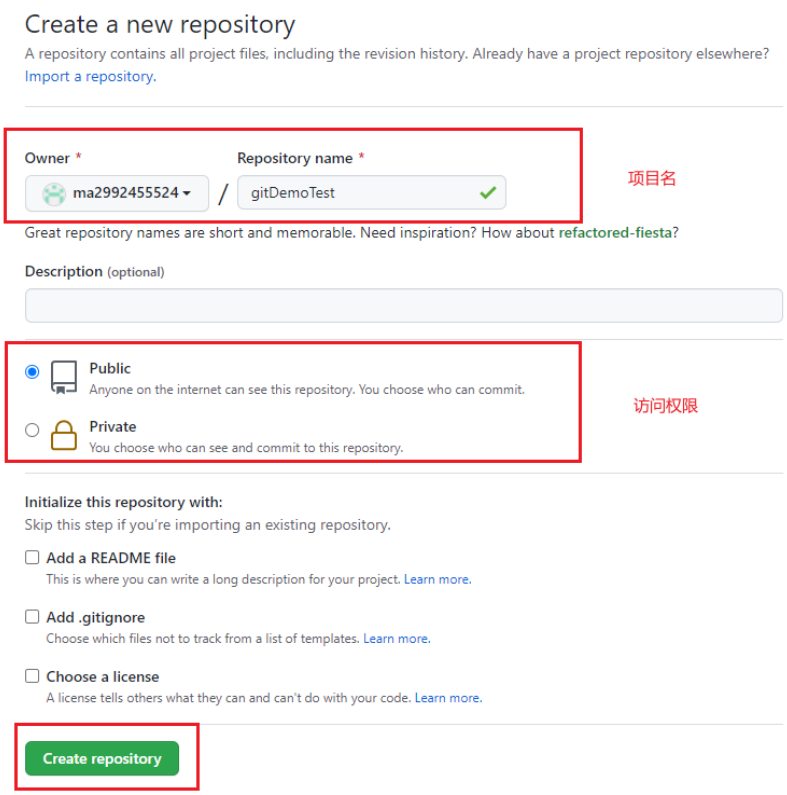


3、邮箱验证

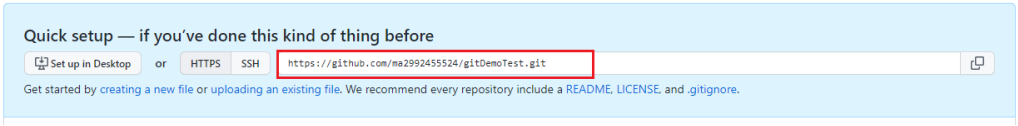



Search or jump to...
Pull requests
Issues
Marketplace
Explore
Repositories
New
Find a repository...
ma2992455524/mjs
ma2992455524/majiasheng.github.io
Recent activity
When you take actions across GitHub, we'll provide links to that activity here.
Learn Git and GitHub without any code!
Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.
Read the guide
Start a project
Universe 2021
Watch the sessions on demand
New repository
Import repository
New gist
New organization
New project
All activity
Introduce yourself
The easiest way to introduce yourself on GitHub is by creating a README in a repository about you! You can start here:
Dismiss this
Continue
Discover interesting projects and people to populate your personal news feed.
Your news feed helps you keep up with recent activity on repositories you watch or star and people you follow.
Explore GitHub

4、创建项目


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 *
ma2992455524
gitDemoTest
项目名
Great repository names are short and memorable. Need inspiration? How about refactored-fiesta?
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.
访问权限
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.
Choose a license
A license tells others what they can and can't do with your code. Learn more.
Create repository

5、获取 http 连接



6、使用 git 起别名

Git remote add 别名 github 连接

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git remote add gitDemoTest https://github.com/ma2992455524/gitDemoTest.git
```

7、查看别名

Git remote -v

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git remote -v
gitDemoTest     https://github.com/ma2992455524/gitDemoTest.git (fetch)
gitDemoTest     https://github.com/ma2992455524/gitDemoTest.git (push)

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

一个用于上传,一个用于拉去

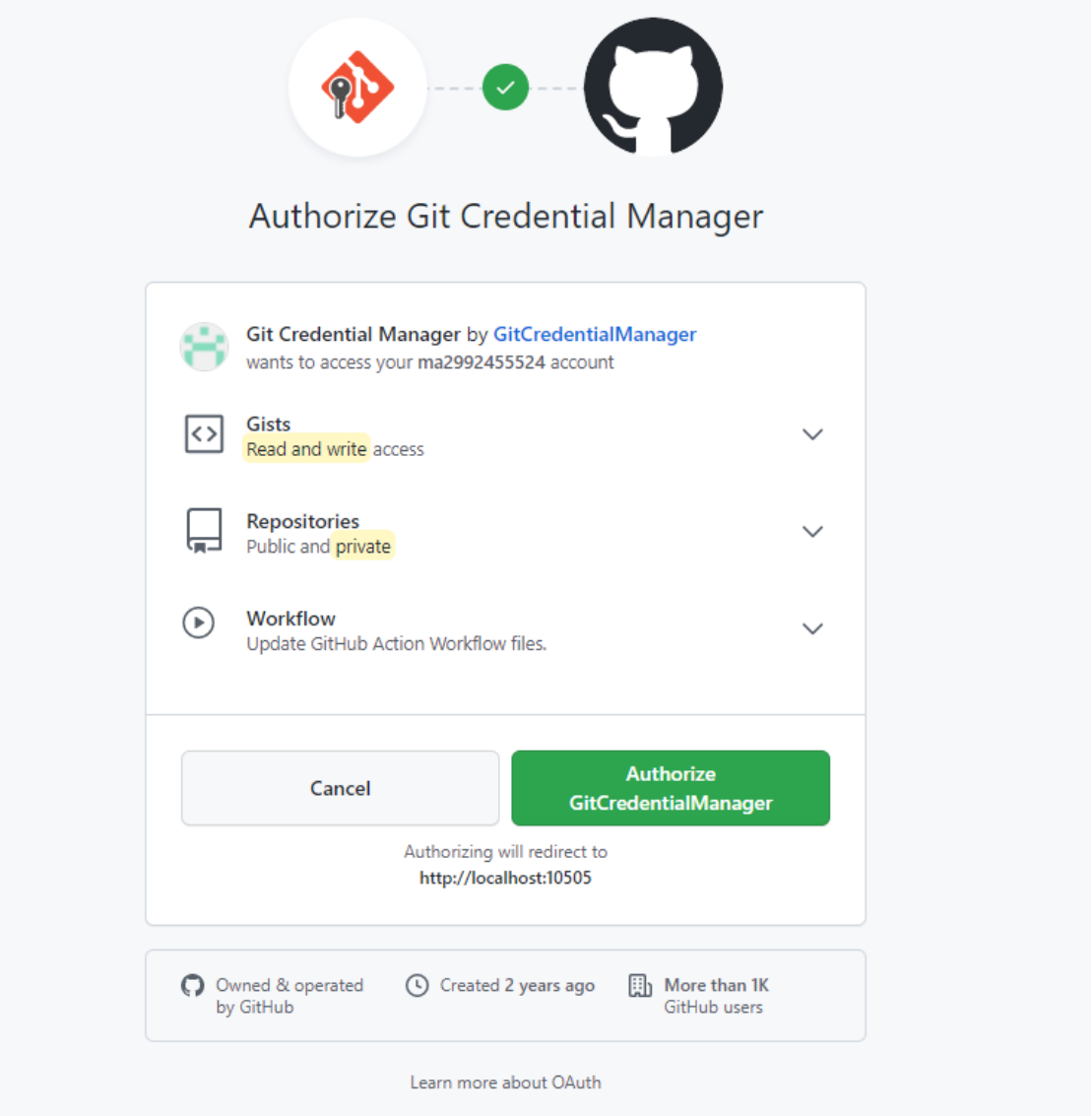
(二)、上传远程仓库(最小分支单位)

1、上传

Git push 别名(连接名) 分支名

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git push gitDemoTest master
```

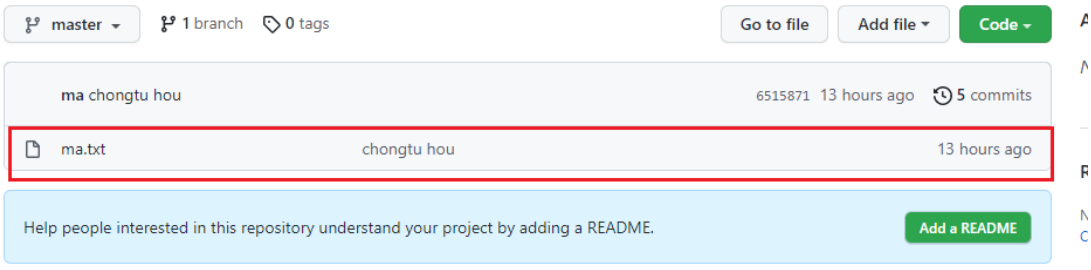
2、使用浏览器登录


Authorize Git Credential Manager
Git Credential Manager by GitCredentialManager
wants to access your ma2992455524 account
Gists
Read and write access
Repositories
Public and private
Workflow
Update GitHub Action Workflow files.
Cancel
Authorize GitCredentialManager
Authorizing will redirect to
http://localhost:10505
Owned & operated by GitHub
Created 2 years ago
More than 1K GitHub users
Learn more about OAuth

3、上传成功

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git push gitDemoTest master
Enumerating objects: 15, done.
Counting objects: 100% (15/15), done.
Delta compression using up to 12 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (15/15), 1.09 KiB | 560.00 KiB/s, done.
Total 15 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/ma2992455524/gitDemoTest.git
 * [new branch]      master -> master

aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$
```

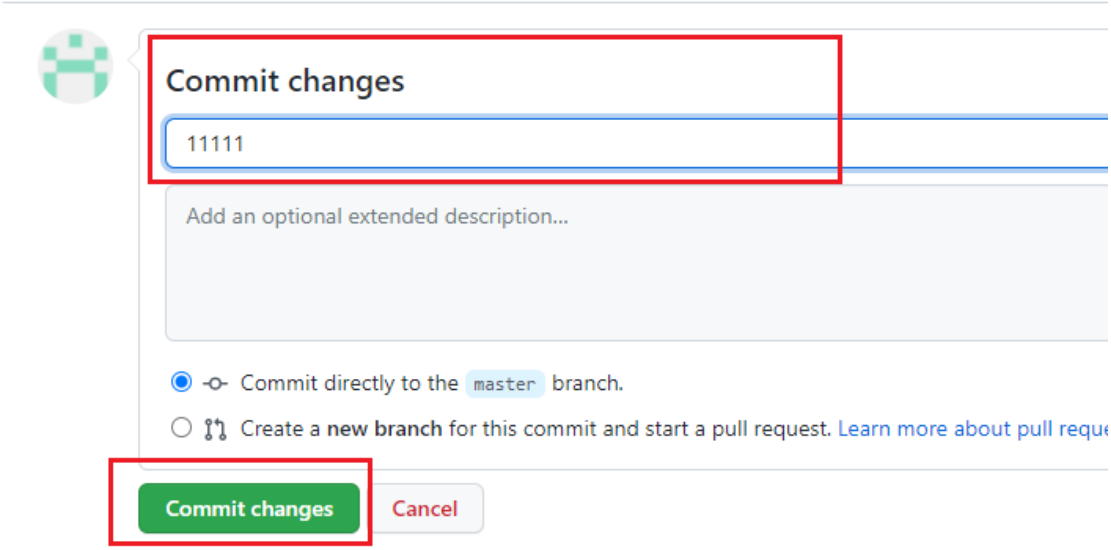
4、验证



(三)、拉取远程仓库到本地仓库

1、修改远程仓库的文件

```
6   hello majaisheng!
7   hello majaisheng!
8   hello majaisheng!
9   hello majaisheng!
10  hello majaisheng! 2222
11  hello majaisheng!
12  hello majaisheng! 333333
13  1111111111111111
```



2、本地仓库拉取

Git pull 别名(连接名) 拉取的分支名

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ git pull gitDemoTest master
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 645 bytes | 2.00 KiB/s, done.
From https://github.com/ma2992455524/gitDemoTest
 * branch            master     -> FETCH_HEAD
   6515871..338eaae  master     -> gitDemoTest/master
Updating 6515871..338eaae
Fast-forward
 ma.txt | 1 +
 1 file changed, 1 insertion(+)
```

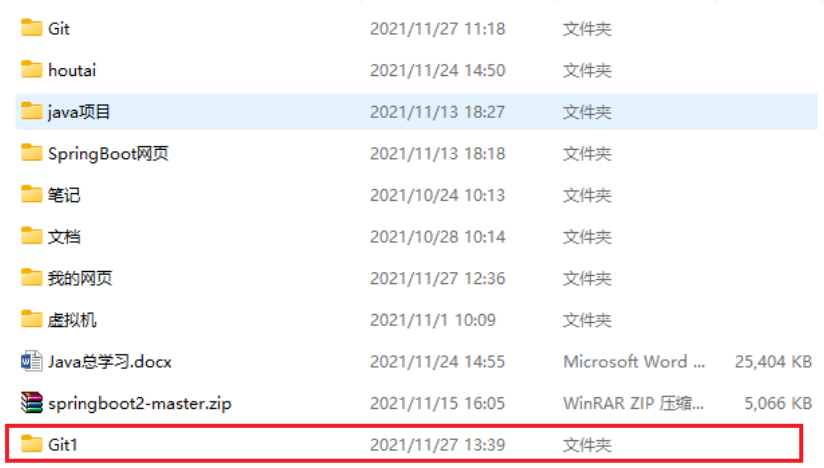
3、查看文件

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git (master)
$ cat ma.txt
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng! 2222
hello majaisheng!
hello majaisheng! 333333
1111111111111111
```

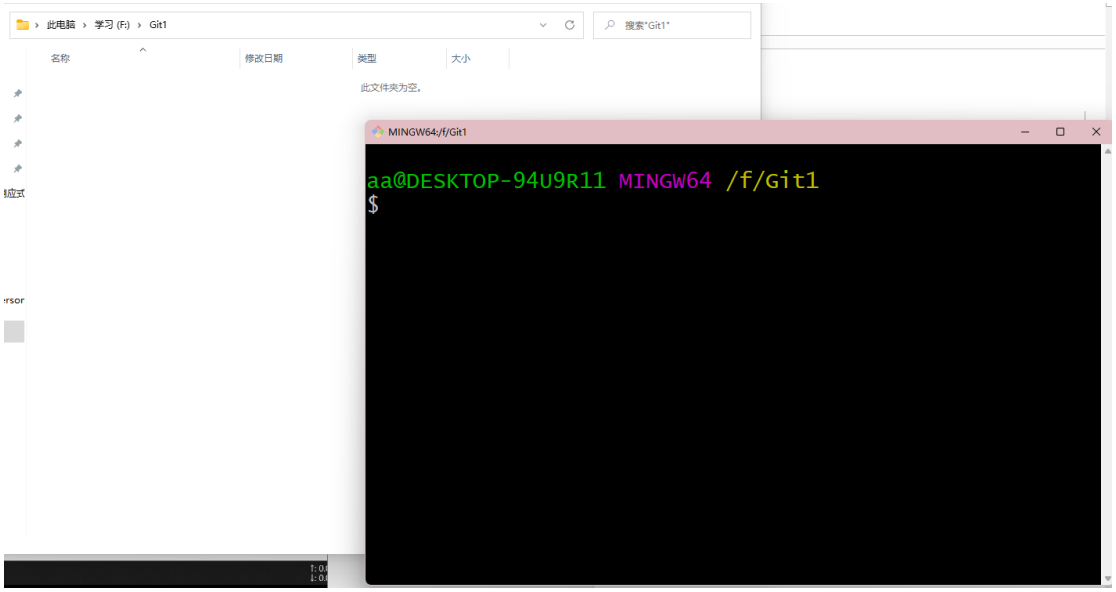
(四)、克隆远程仓库到本地

克隆仓库做了三件事
1:克隆文件
2:创建本地仓库
3:给连接起别名

1、新建一个文件



2、进入文件打开 git



3、克隆远程仓库

Git clone 连接

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git1
$ git clone https://github.com/ma2992455524/gitDemoTest.git
Cloning into 'gitDemoTest'...
remote: Enumerating objects: 18, done.
remote: Counting objects: 100% (18/18), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 18 (delta 3), reused 14 (delta 2), pack-reused 0
Receiving objects: 100% (18/18), done.
Resolving deltas: 100% (3/3), done.

aa@DESKTOP-94U9R11 MINGW64 /f/Git1
```

4、查看文件

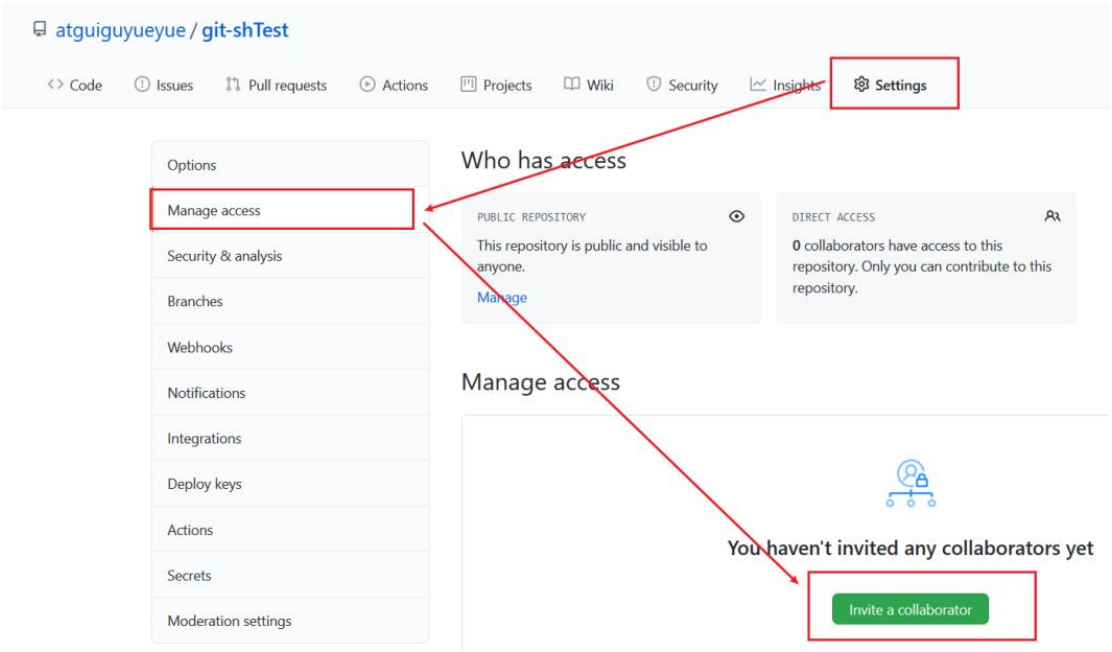
```
aa@DESKTOP-94U9R11 MINGW64 /f/Git1
$ dir
gitDemoTest

aa@DESKTOP-94U9R11 MINGW64 /f/Git1
$ cd gitDemoTest/

aa@DESKTOP-94U9R11 MINGW64 /f/Git1/gitDemoTest (master)
$ dir
ma.txt

aa@DESKTOP-94U9R11 MINGW64 /f/Git1/gitDemoTest (master)
$ cat ma.txt
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng!
hello majaisheng! 2222
hello majaisheng!
hello majaisheng! 333333
1111111111111111

aa@DESKTOP-94U9R11 MINGW64 /f/Git1/gitDemoTest (master)
$
```

5、查看别名

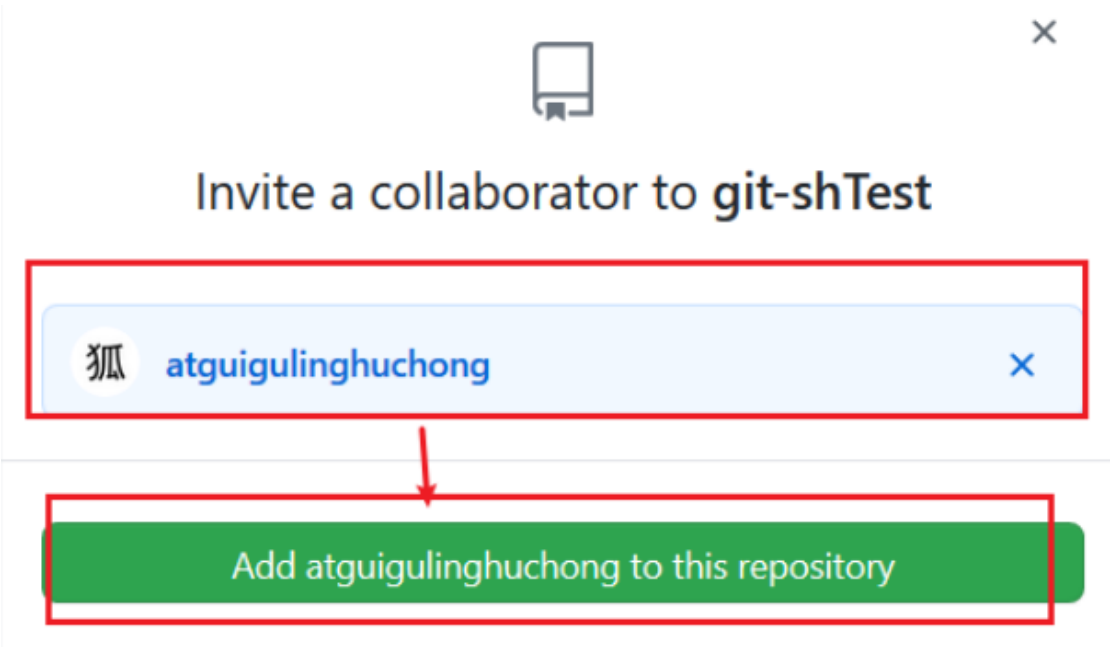
```
aa@DESKTOP-94U9R11 MINGW64 /f/Git1/gitDemoTest (master)
$ git remote -v
origin  https://github.com/ma2992455524/gitDemoTest.git (fetch)
origin  https://github.com/ma2992455524/gitDemoTest.git (push)

aa@DESKTOP-94U9R11 MINGW64 /f/Git1/gitDemoTest (master)
$
```

(五)、团队协作

1、邀请别人加入你的团队

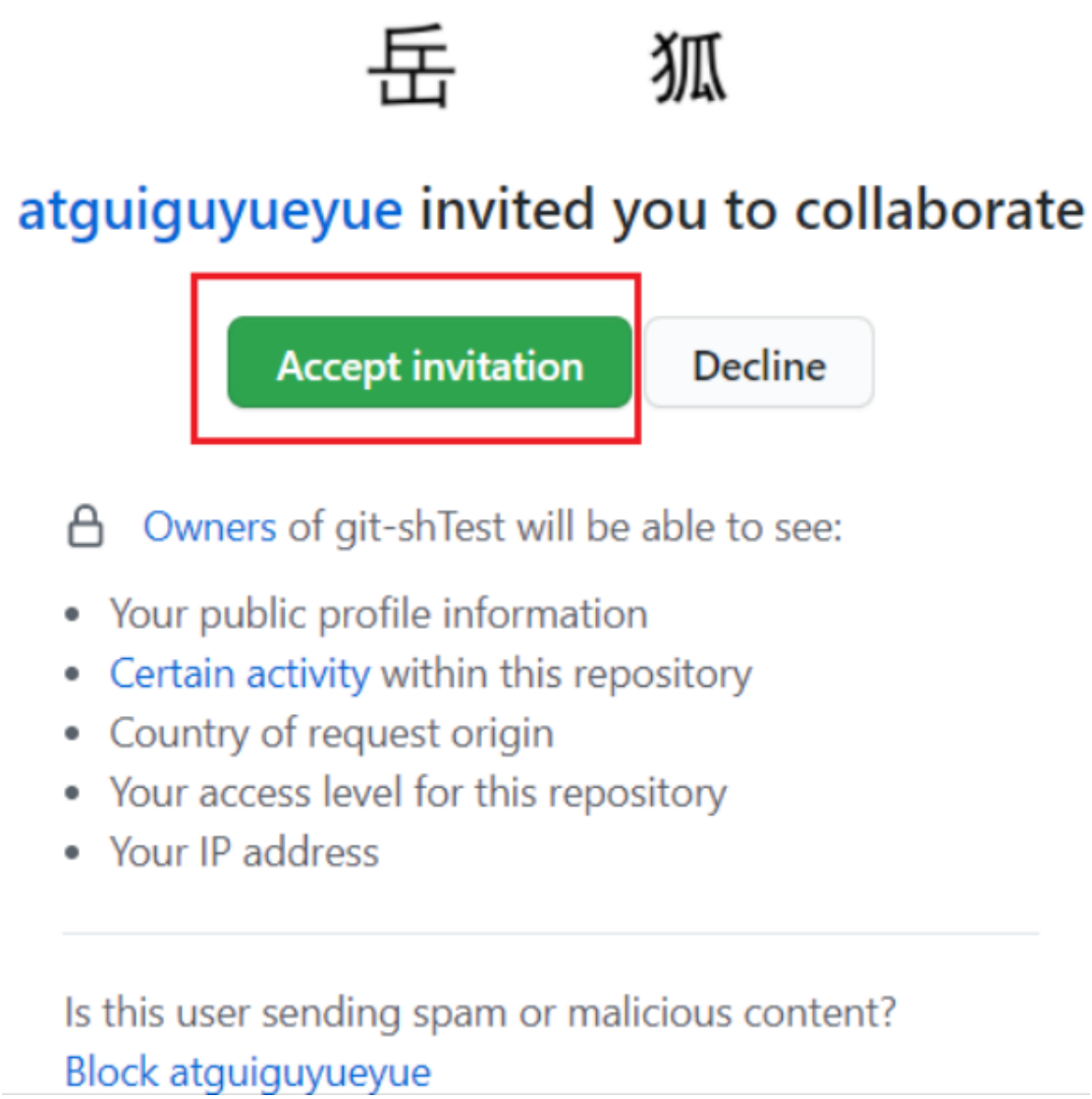


2、填入想要合作的人

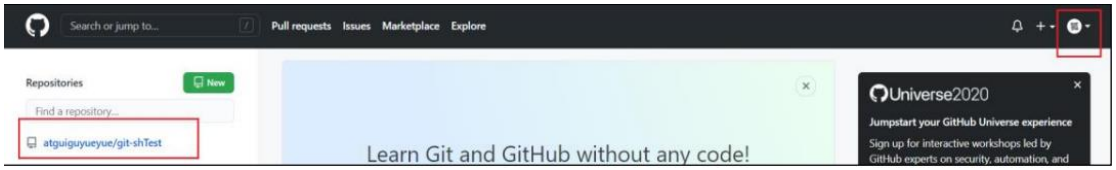


3、复制地址并通过微信等方式发送给该用户

4、在被邀请账号中的地址栏复制收到邀请的链接，点击接受邀请



5、成功之后可以在被邀请这个账号上看到 git-Test 的远程仓库



6、邀请账号修改上传文件

vim hello.txt

```
$ cat hello.txt
hello git! hello atguigu! 2222222222222
hello git! hello atguigu! 33333333333333
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu! 我是最帅的，比岳不群还帅
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu! master test
hello git! hello atguigu! hot-fix test
```

7、上传缓存区—本地仓库—远程仓库

```
$ git add hello.txt
--将暂存区的文件上传到本地库
Layne@LAPTOP-Layne   MINGW64   /d/Git-Space/pro-linghuchong/git-shTest
(master)
$ git commit -m "lhc commit" hello.txt
[master 5dabe6b] lhc commit
 1 file changed, 1 insertion(+), 1 deletion(-)
--将本地库的内容 push 到远程仓库
Layne@LAPTOP-Layne   MINGW64   /d/Git-Space/pro-linghuchong/git-shTest
(master)
$ git push origin master
Logon failed, use ctrl+c to cancel basic credential prompt.
Username for 'https://github.com': atguigulinghuchong
Counting objects: 3, done.
Delta compression using up to 12 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 309 bytes | 309.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/atguiguyueyue/git-shTest.git
   7cb4d02..5dabe6b  master -> master
```

8、主用户拉取文件

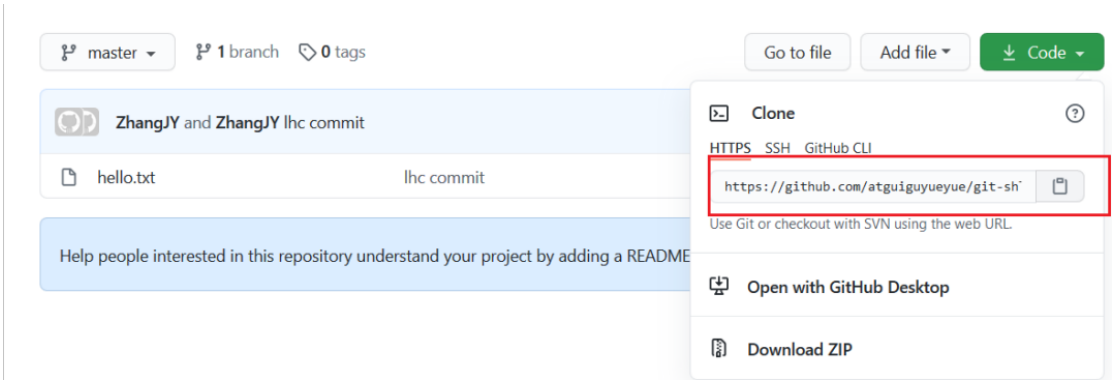
```
$ git pull ori master
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (1/1), done.
remote: Total 3 (delta 1), reused 3 (delta 1), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/atguiguyueyue/git-shTest
 * branch            master     -> FETCH_HEAD
   7cb4d02..5dabe6b  master     -> ori/master
Updating 7cb4d02..5dabe6b
Fast-forward
 hello.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

9、查看文件

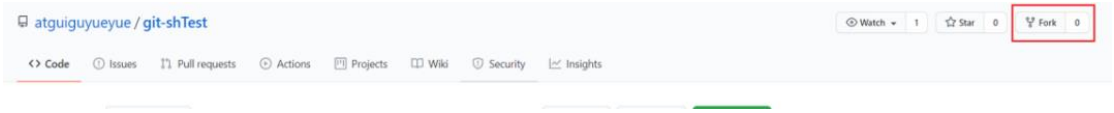
```
Layne@LAPTOP-Layne MINGW64 /d/Git-Space/SH0720 (master)
$ cat hello.txt
hello git! hello atguigu! 2222222222222
hello git! hello atguigu! 33333333333333
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu! 我是最帅的，比岳不群还帅
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu!
hello git! hello atguigu! master test
hello git! hello atguigu! hot-fix test
```

(六)、跨团队协作

1、将自己仓库地址发给别人

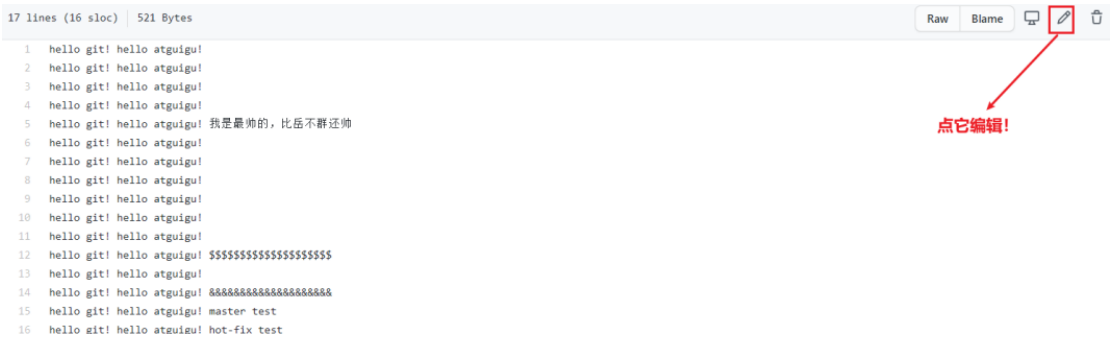


2、使用别人账号打开网址点击 fork 叉到自己仓库

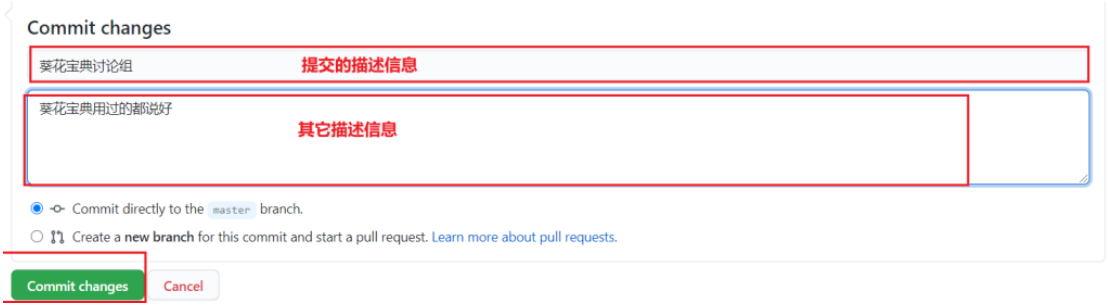


3、插入后查看信息

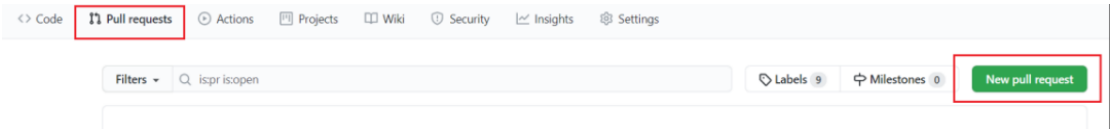
4、别人就可以编辑文件

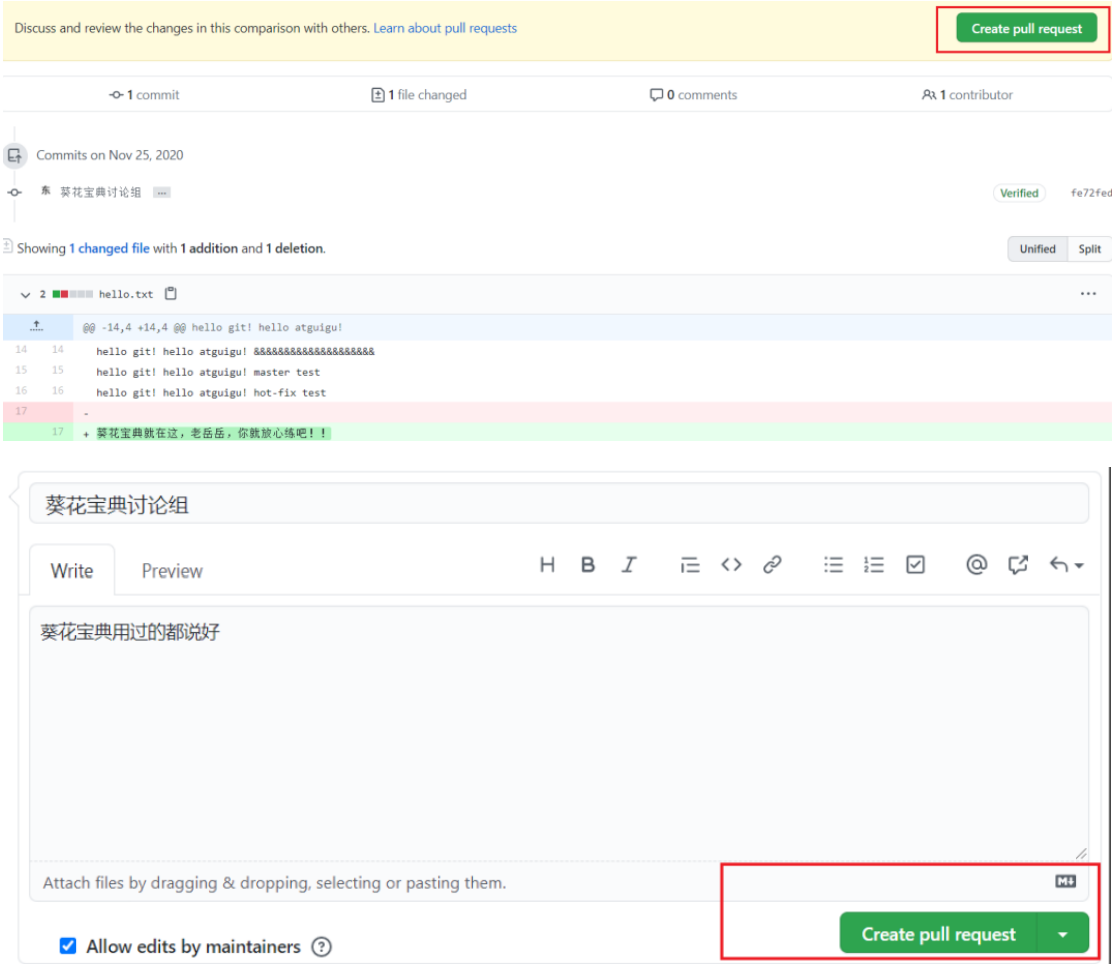


5、编辑后填入信息提交

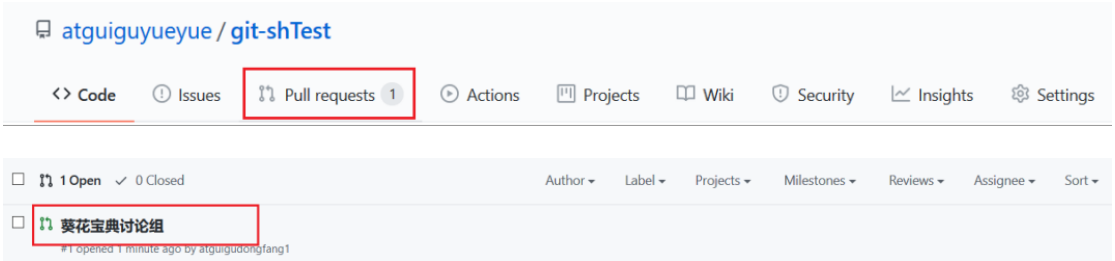


6、点击 pull 请求创建一个新的请求

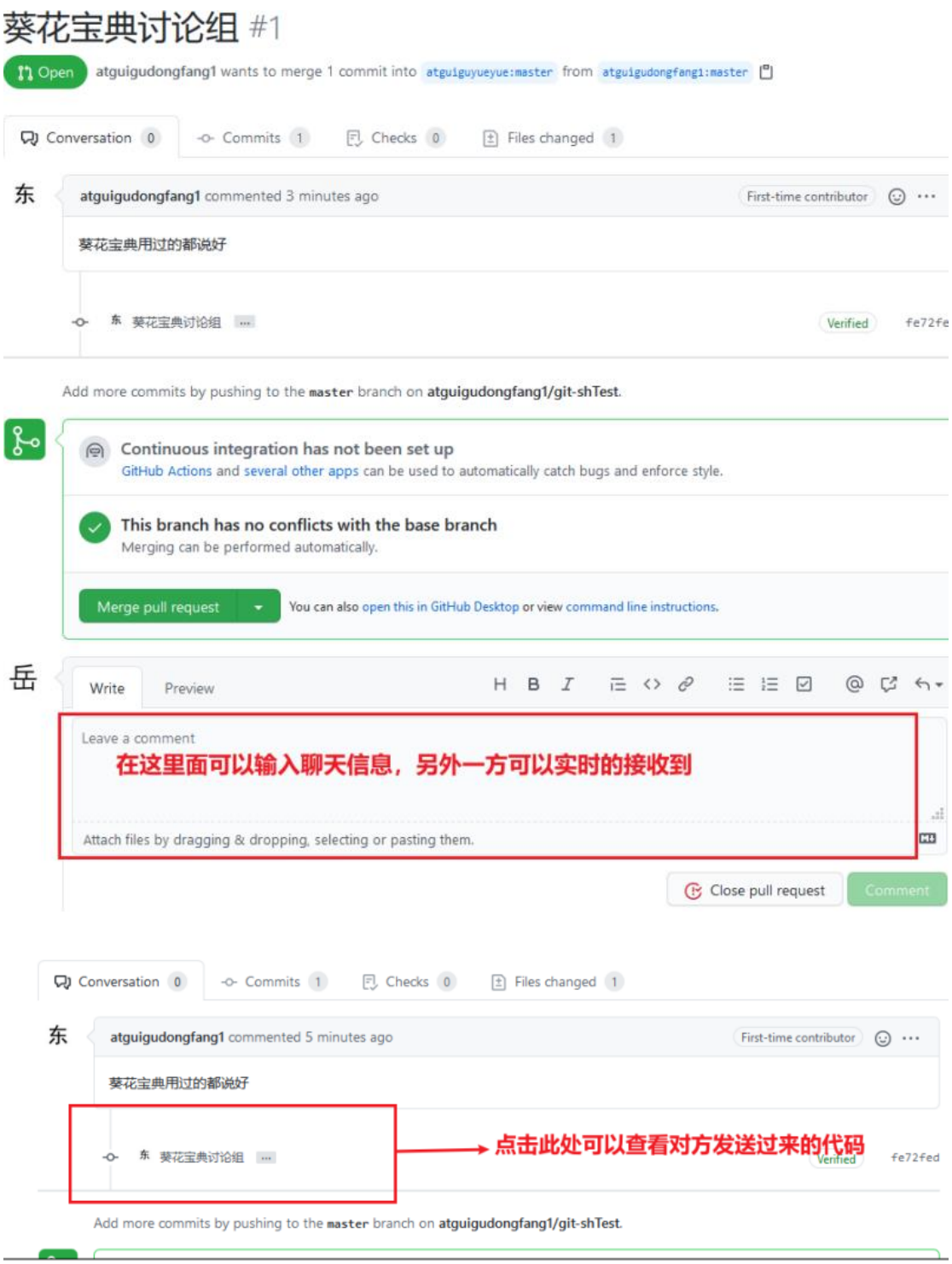




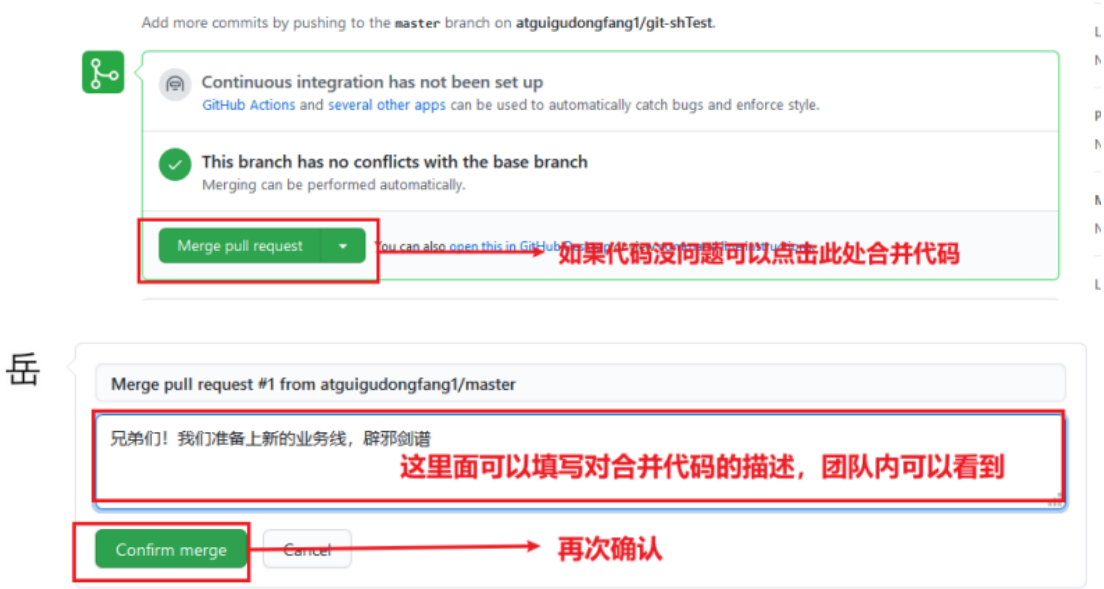
7、回到以前的账号就发现一个 pull request 请求



8、可以进入小组进行讨论

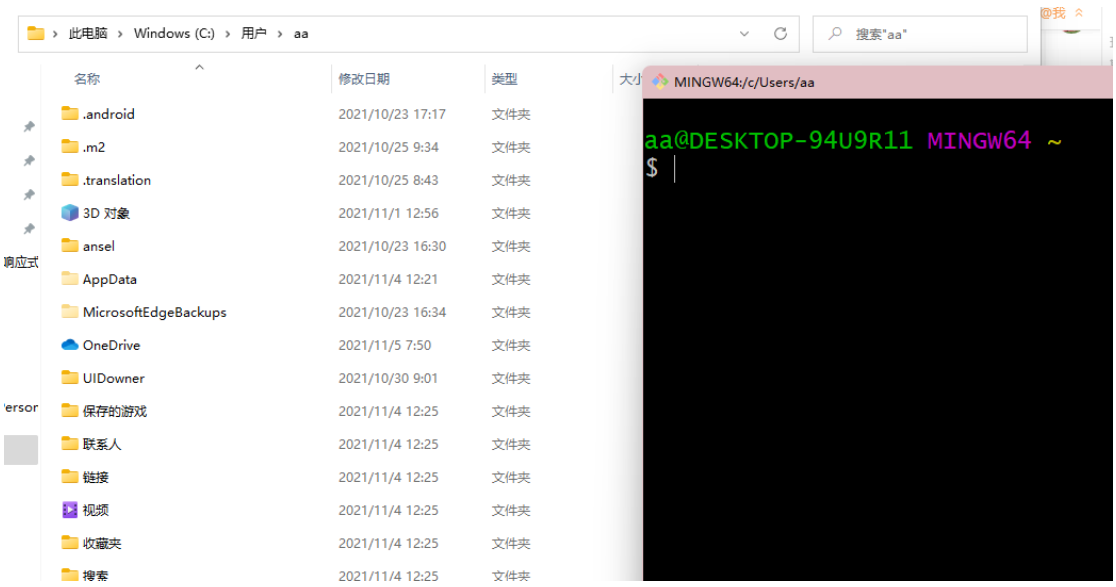


9、如果代码没问题点击 Merge pull reque 合并



(七)、SSH 免密登录

1、进入家目录打开 git



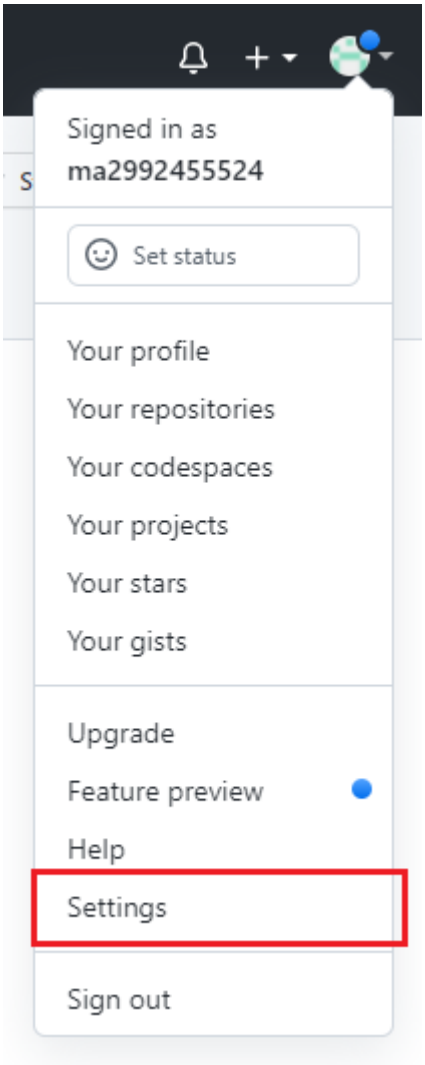
2、创建私钥

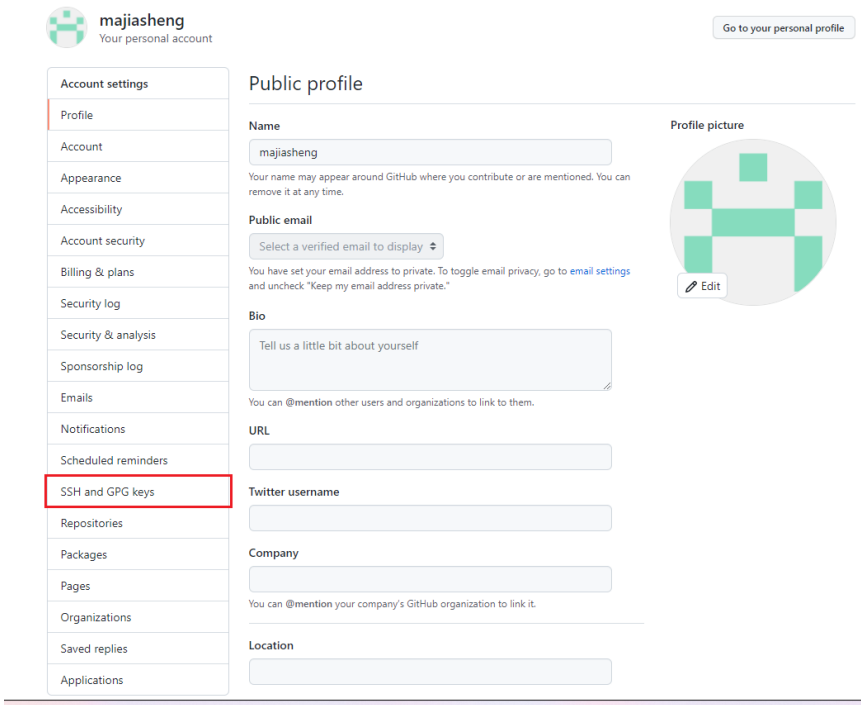
```
aa@DESKTOP-94U9R11 MINGW64 ~
$  ssh-keygen -t rsa -C ma2992455524
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/aa/.ssh/id_rsa):
Created directory '/c/Users/aa/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/aa/.ssh/id_rsa
Your public key has been saved in /c/Users/aa/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:7ZZoi8+b7sjznPoAqh11xrs6a8CipNVA6697yD2R5Jo ma2992455524
The key's randomart image is:
+---[RSA 3072]----+
|                 |
|  .              |
| . .             |
|  o ..    .      |
| o =o.+ S .      |
|..=o=+ . o .     |
|++oB .o o +      |
|ooE.B..O =       |
|. +=o=*O&.       |
+----[SHA256]-----+

aa@DESKTOP-94U9R11 MINGW64 ~
$
```

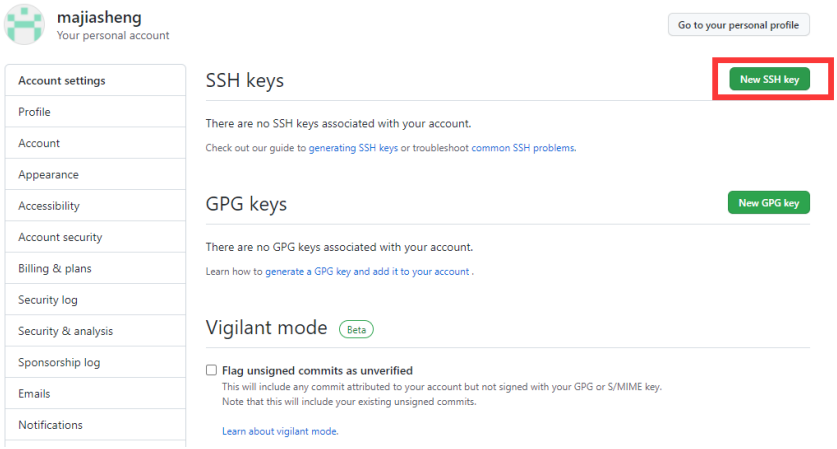
3、查看私钥

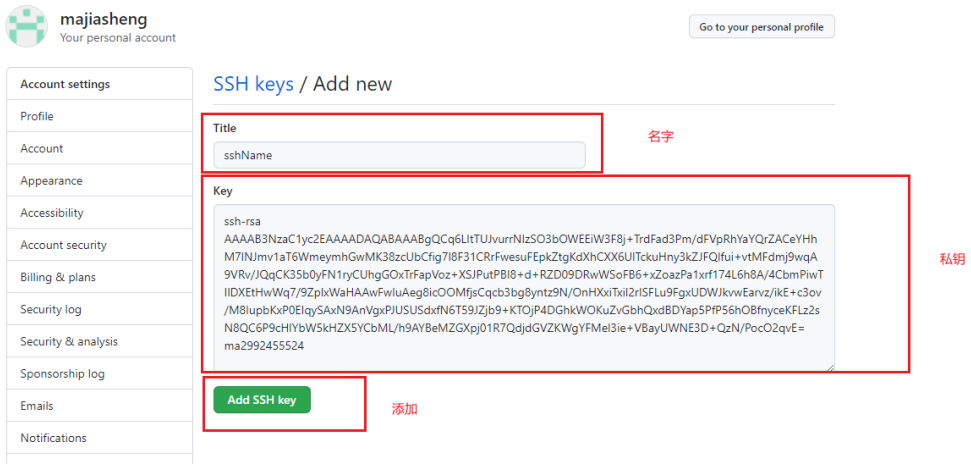
```
aa@DESKTOP-94U9R11 MINGW64 ~/.ssh
$ cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQCq6LItTUJvurrNIzSO3bOWEEiW3F8j+TrdFad3Pm/dFVpRhYaYQrZACeYHhM7lNJmv1aT6WmeymhGWMK38zcUbCfig7l8F31C
RrFwesuFEpkZtgKdXhCXX6UlTckuHny3kZJFQlfui+vtMFdmj9wqA9VRv/JQqCK35bOyFN1ryCUhgGOxTrFapVoz+XSJPutPBI8+d+RZDO9DRWWSoFB6+xZoazPa1xrf174L6h8
A/4CbmPiwTIlDXEtHWWq7/9ZplxWaHAAwFwIuAeg8icOOMfjsCqcb3bg8yntz9N/OnHXxiTxil2rISFLu9FgxUDWJkvwEarvz/ikE+c3ov/M8IupbKxPOEIqySAxN9AnVgxPJUS
UsdxfN6T59JZjb9+KTOjP4DGhkWOKuZvGbhQxdBDYap5PfP56hOBfnyceKFLz2sN8QC6P9cHlYbW5kHZX5YCbML/h9AYBeMZGXpj01R7QdjdGVZKWgYFMe13ie+VBayUWNE3D+Q
zN/PocO2qvE= ma2992455524

aa@DESKTOP-94U9R11 MINGW64 ~/.ssh
$
```

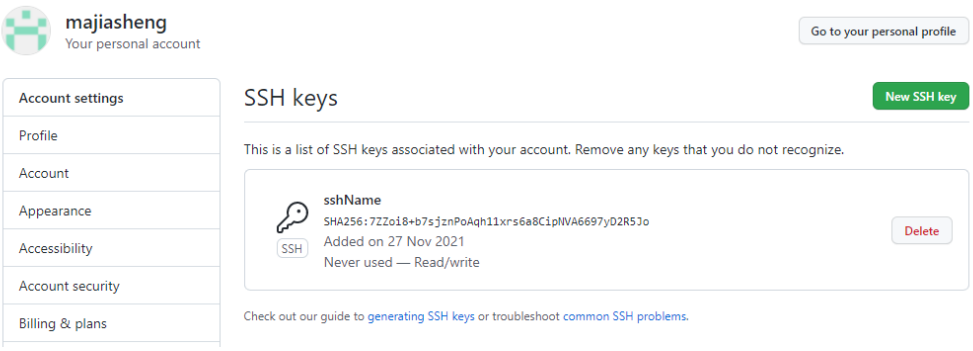
4、应用到 github 账户上



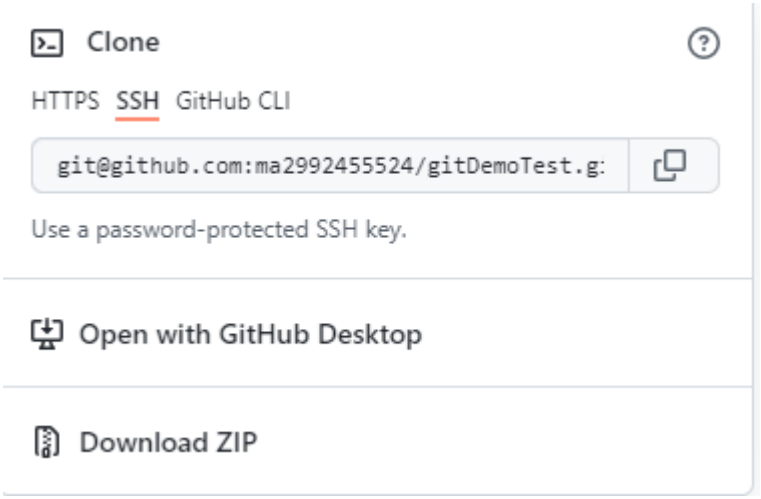

majiasheng
Your personal account
Go to your personal profile
Account settings
Profile
Account
Appearance
Accessibility
Account security
Billing & plans
Security log
Security & analysis
Sponsorship log
Emails
Notifications
Scheduled reminders
SSH and GPG keys
Repositories
Packages
Pages
Organizations
Saved replies
Applications
Public profile
Name
majiasheng
Your name may appear around GitHub where you contribute or are mentioned. You can remove it at any time.
Public email
Select a verified email to display
You have set your email address to private. To toggle email privacy, go to email settings and uncheck "Keep my email address private."
Bio
Tell us a little bit about yourself
You can @mention other users and organizations to link to them.
URL
Twitter username
Company
You can @mention your company's GitHub organization to link it.
Location
Profile picture
Edit


majiasheng
Your personal account
Go to your personal profile
Account settings
Profile
Account
Appearance
Accessibility
Account security
Billing & plans
Security log
Security & analysis
Sponsorship log
Emails
Notifications
SSH keys
New SSH key
There are no SSH keys associated with your account.
Check out our guide to generating SSH keys or troubleshoot common SSH problems.
GPG keys
New GPG key
There are no GPG keys associated with your account.
Learn how to generate a GPG key and add it to your account .
Vigilant mode
Beta
Flag unsigned commits as unverified
This will include any commit attributed to your account but not signed with your GPG or S/MIME key.
Note that this will include your existing unsigned commits.
Learn about vigilant mode.


majiasheng
Your personal account
Go to your personal profile
Account settings
Profile
Account
Appearance
Accessibility
Account security
Billing & plans
Security log
Security & analysis
Sponsorship log
Emails
Notifications
SSH keys / Add new
Title
sshName
名字
Key
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQCq6LItTUJvurrNIzSO3bOWEEiW3F8j+TrdFad3Pm/dFVpRhYaYQrZACeYHh
M7INJmv1aT6WmeymhGwMK38zcUbCfig7l8F31CRrFwesuFEpkZtgKdXhCXX6UITckuHny3kZJFQIfui+vtMFdmj9wqA
9VRv/JQqCK35b0yFN1ryCUhgGOxTrFapVoz+XSJPutPBl8+d+RZD09DRwWSoFB6+xZoazPa1xrf174L6h8A/4CbmPiwT
IIDXEtHwWq7/9ZplxWaHAAwFwIuAeg8icOOMfjsCqcb3bg8yntz9N/OnHXxiTxil2rISFLu9FgxUDWJkvwEarvz/ikE+c3ov
/M8IupbKxP0EIqySAxN9AnVgxPJUSUSdxfN6T59JZjb9+KTOjP4DGhkWOKuZvGbhQxdBDYap5PfP56hOBfnyceKFLz2s
N8QC6P9cHIYbW5kHZX5YCbML/h9AYBeMZGXpj01R7QdjdGVZKWgYFMel3ie+VBayUWNE3D+QzN/PocO2qvE=
ma2992455524
私钥
Add SSH key
添加



5、复制 ssh 连接



6、打开 git 上传测试

```
aa@DESKTOP-94U9R11 MINGW64 /f/Git1/gitDemoTest (master)
$ git push git@github.com:ma2992455524/gitDemoTest.git master
The authenticity of host 'github.com (20.205.243.166)' can't be establishe
d.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvC
OqU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hos
ts.
Everything up-to-date

aa@DESKTOP-94U9R11 MINGW64 /f/Git1/gitDemoTest (master)
$
```

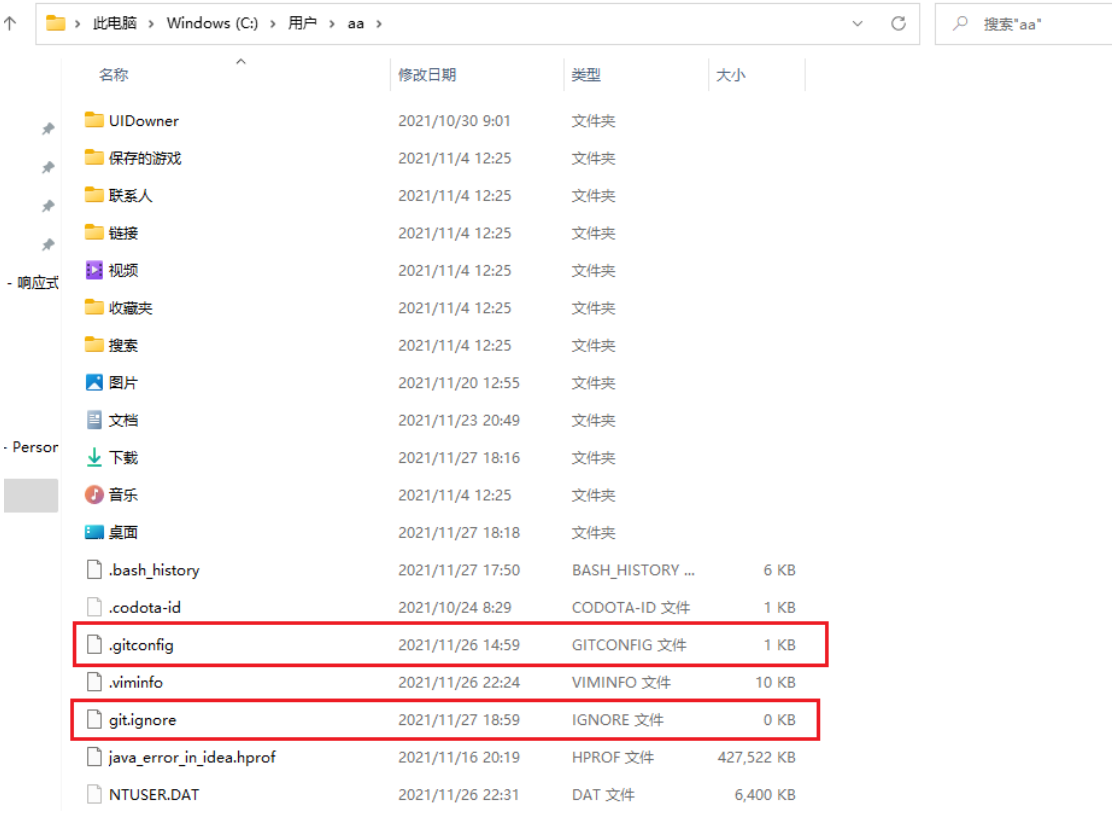
第一次连接是否保存:是

四、IDEA 使用 Git

(一)、配置环境

1、定义配置 git 提交配置文件

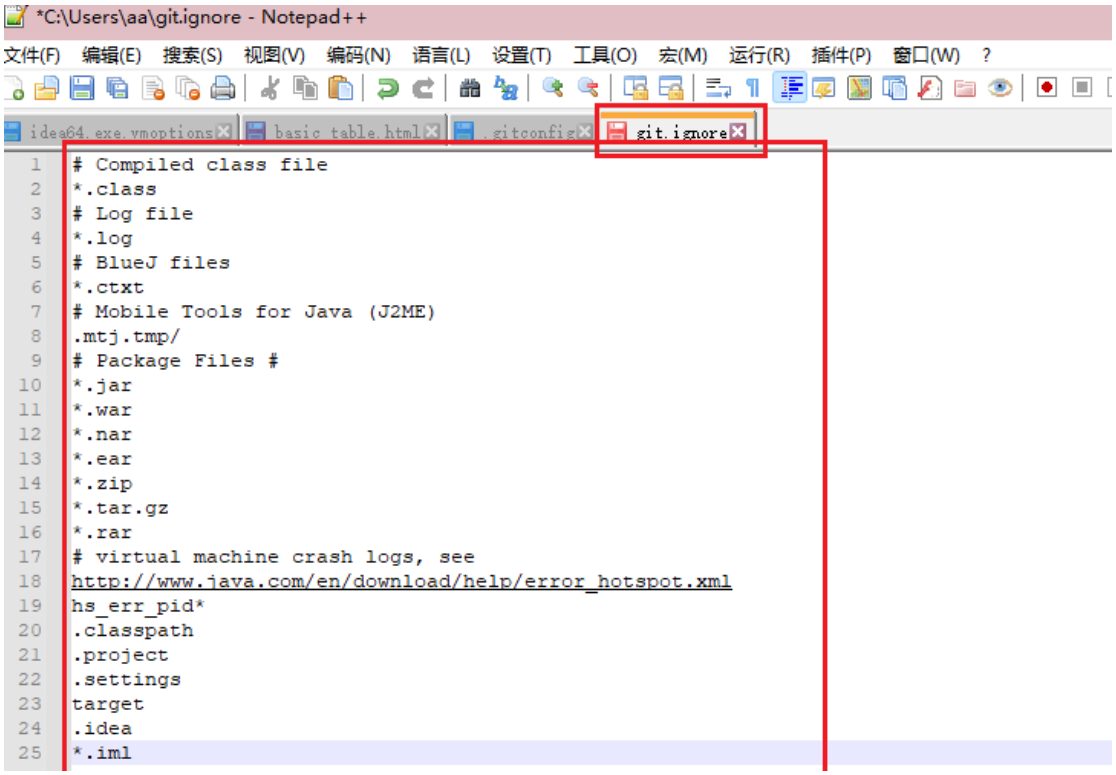
在家目录创建 xxxx.ignore 的文件,和.gitconfig 放在一个目录



2、添加配置

```
# Compiled class file
*.class
# Log file
*.log
# BlueJ files
*.ctxt
```

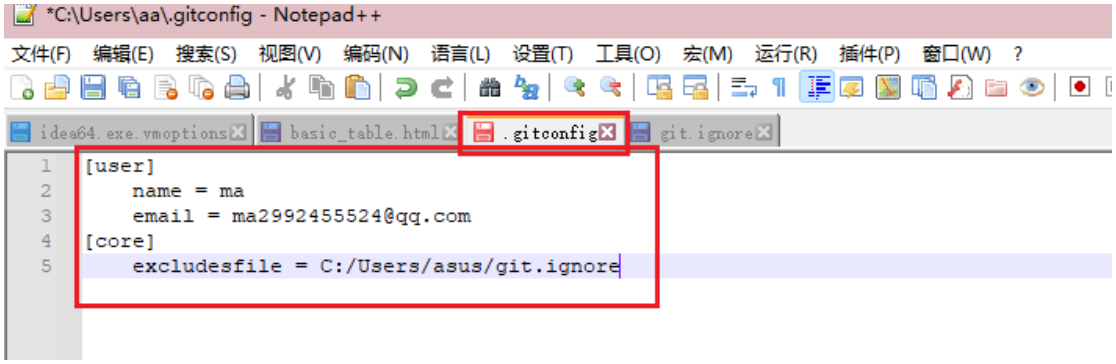
```
# Mobile Tools for Java (J2ME)
.mtj.tmp/
# Package Files #
*.jar
*.war
*.nar
*.ear
*.zip
*.tar.gz
*.rar
# virtual machine crash logs, see
http://www.java.com/en/download/help/error_hotspot.xml
hs_err_pid*
.classpath
.project
.settings
target
.idea
*.iml
```



3、在.gitconfig 中连接这个文件夹

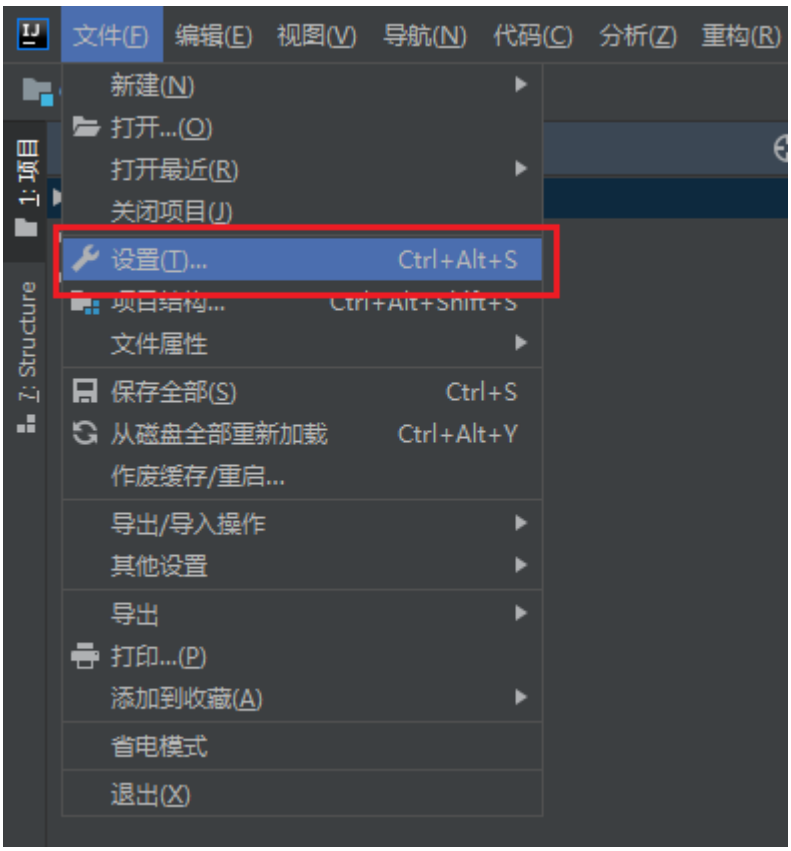
[core]

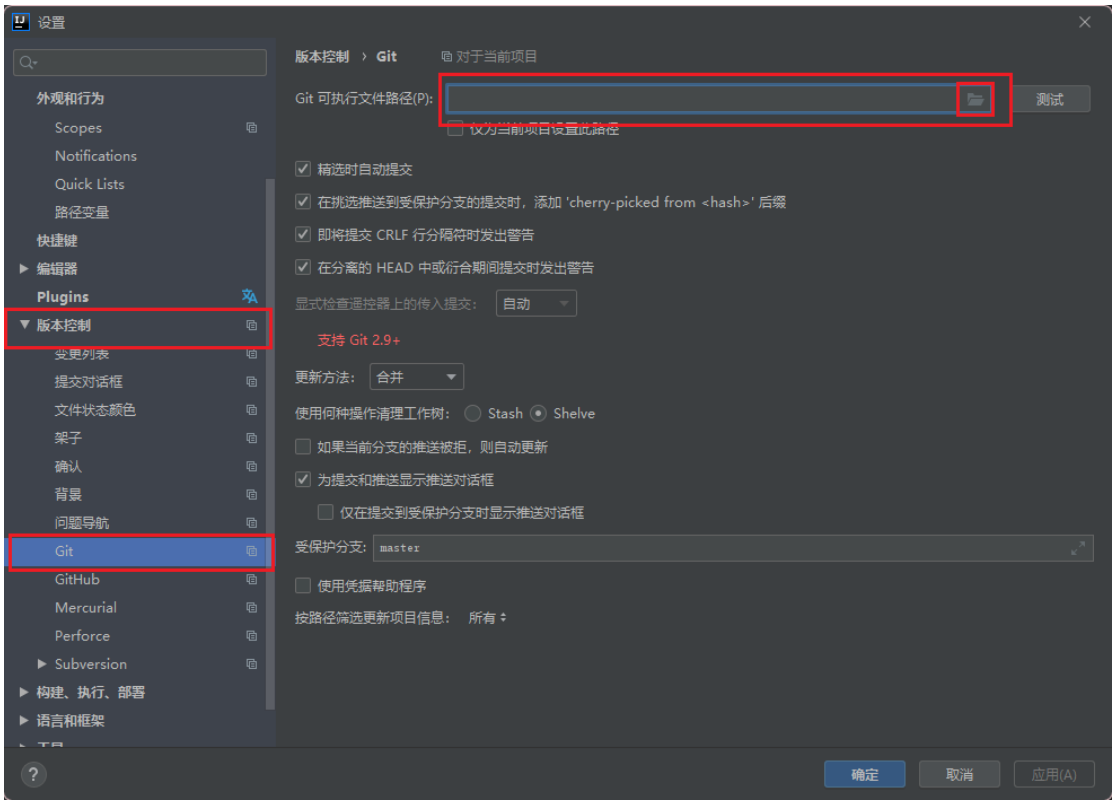
excludesfile = C:/Users/asus/git.ignore

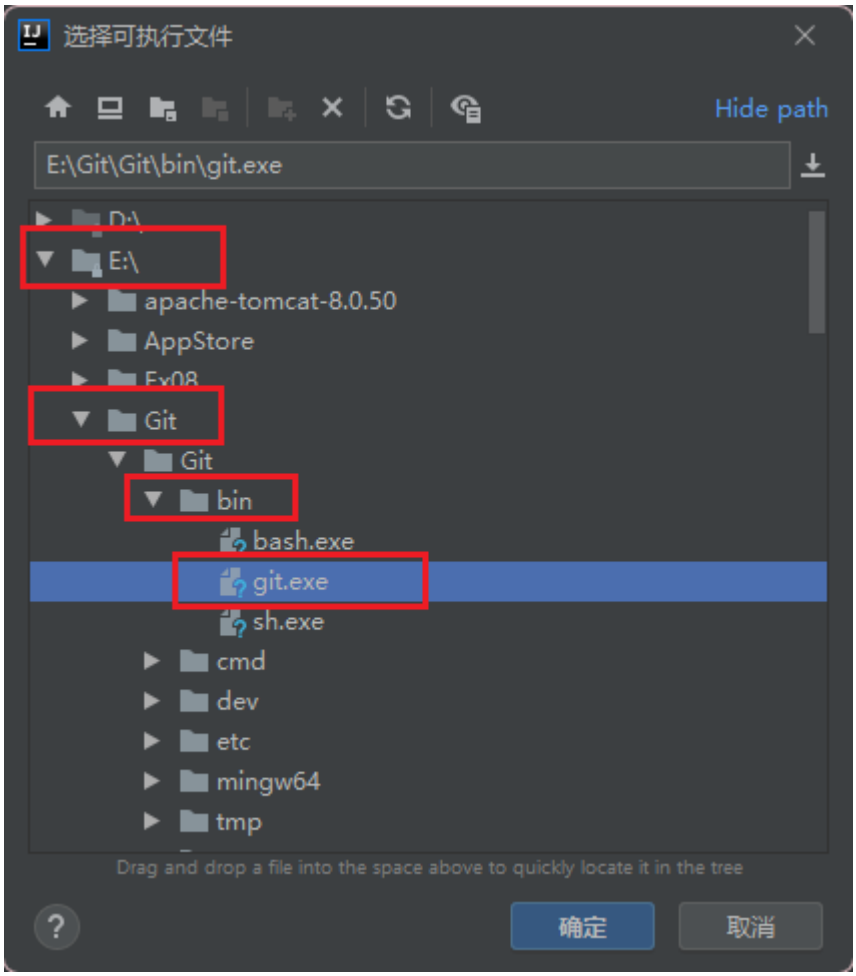


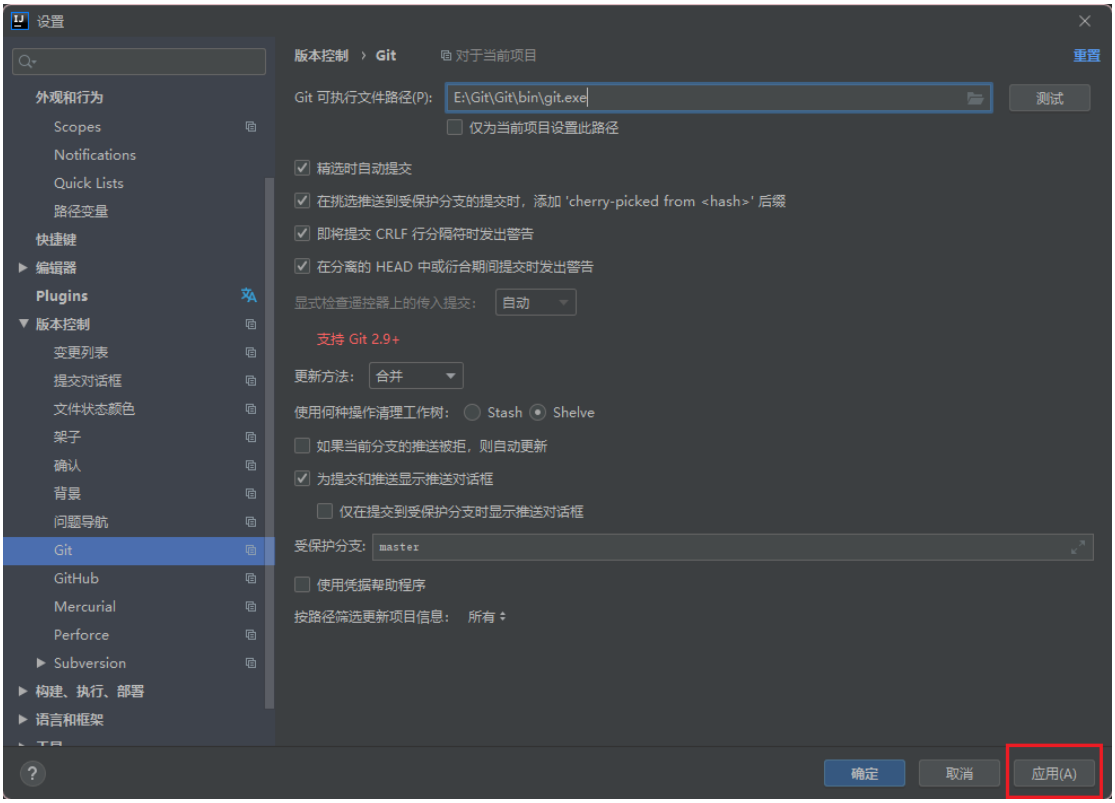
4、IDEA 创建一个普通的 Maven 工程

5、设置 IDEA 的 Git

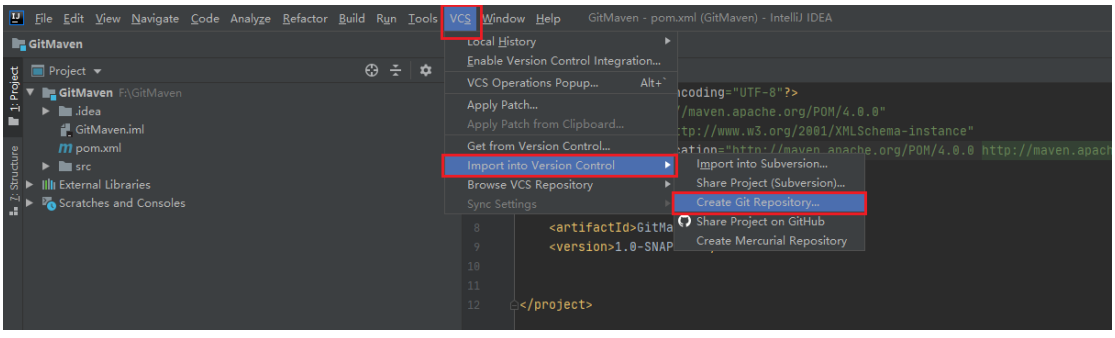


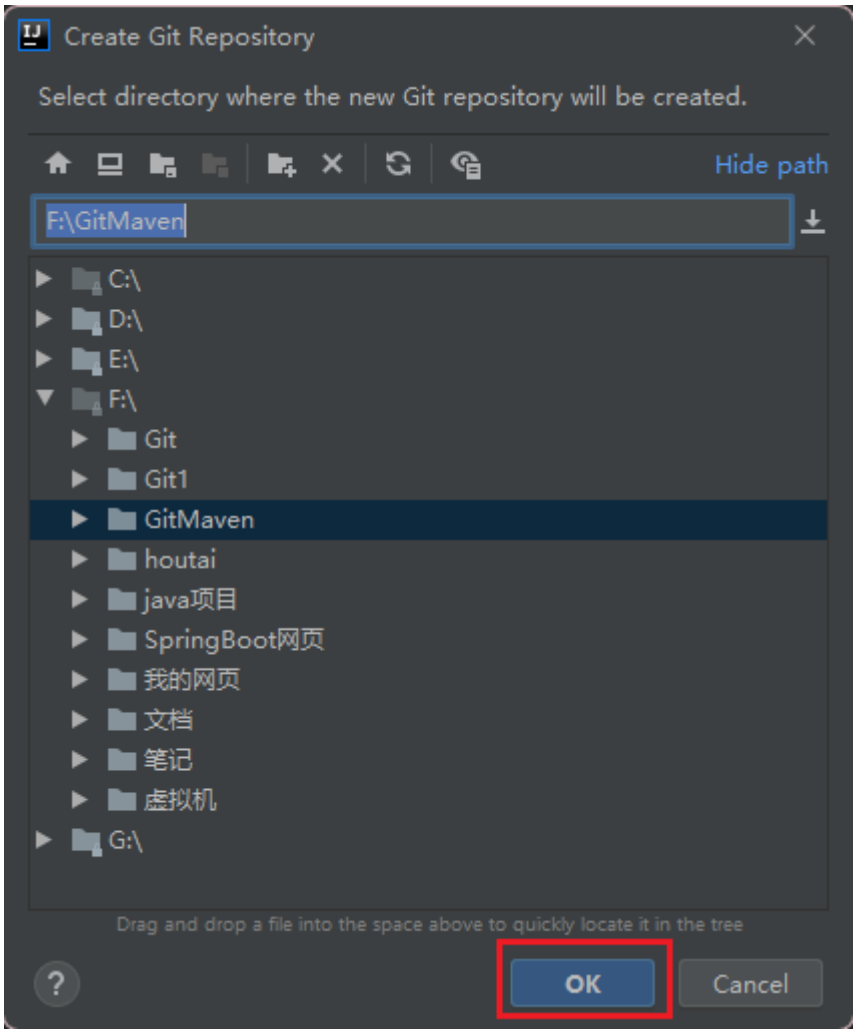

设置
外观和行为
Scopes
Notifications
Quick Lists
路径变量
快捷键
编辑器
Plugins
版本控制
提交对话框
文件状态颜色
架子
确认
背景
问题导航
Git
GitHub
Mercurial
Perforce
Subversion
构建、执行、部署
语言和框架
版本控制 › Git
对于当前项目
Git 可执行文件路径(P):
测试
仅为当前项目设置此路径
精选时自动提交
在挑选推送到受保护分支的提交时，添加 'cherry-picked from <hash>' 后缀
即将提交 CRLF 行分隔符时发出警告
在分离的 HEAD 中或衍合期间提交时发出警告
显式检查遥控器上的传入提交:
自动
支持 Git 2.9+
更新方法:
合并
使用何种操作清理工作树:
Stash
Shelve
如果当前分支的推送被拒，则自动更新
为提交和推送显示推送对话框
仅在提交到受保护分支时显示推送对话框
受保护分支:
master
使用凭据帮助程序
按路径筛选更新项目信息:
所有
确定
取消
应用(A)


选择可执行文件
Hide path
E:\Git\Git\bin\git.exe
D:\
E:\
apache-tomcat-8.0.50
AppStore
Git
Git
bin
bash.exe
git.exe
sh.exe
cmd
dev
etc
mingw64
tmp
Drag and drop a file into the space above to quickly locate it in the tree
确定
取消

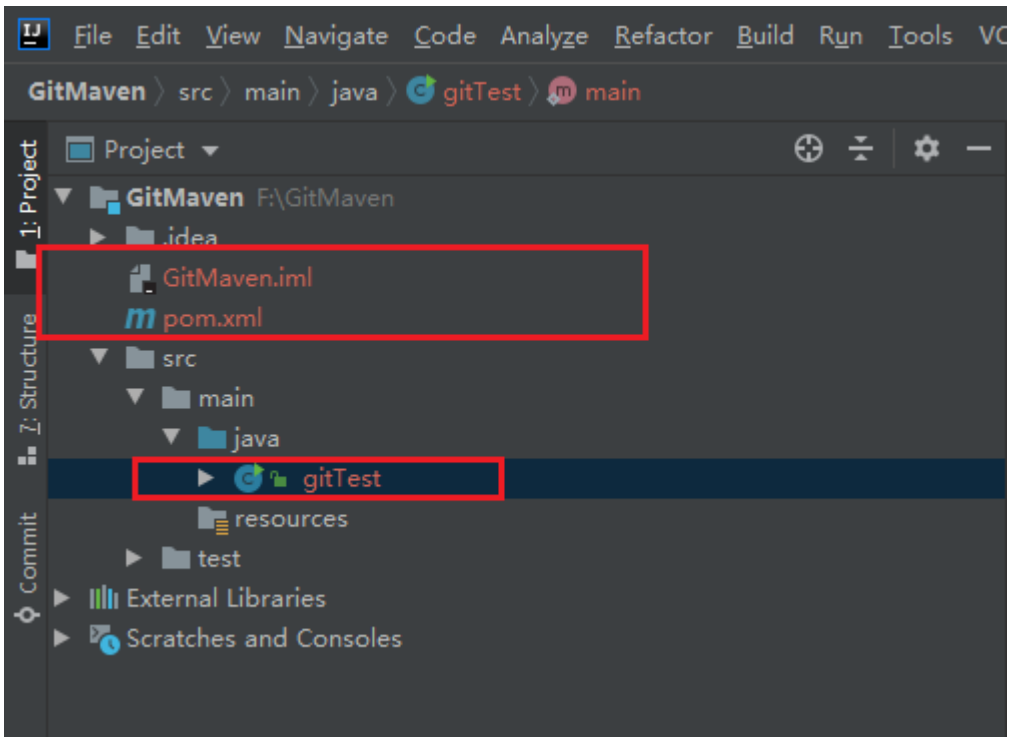

设置
外观和行为
Scopes
Notifications
Quick Lists
路径变量
快捷键
编辑器
Plugins
版本控制
变更列表
提交对话框
文件状态颜色
架子
确认
背景
问题导航
Git
GitHub
Mercurial
Perforce
Subversion
构建、执行、部署
语言和框架
版本控制 › Git
对于当前项目
重置
Git 可执行文件路径(P):
E:\Git\Git\bin\git.exe
测试
仅为当前项目设置此路径
精选时自动提交
在挑选推送到受保护分支的提交时，添加 'cherry-picked from <hash>' 后缀
即将提交 CRLF 行分隔符时发出警告
在分离的 HEAD 中或衍合期间提交时发出警告
显式检查遥控器上的传入提交:
自动
支持 Git 2.9+
更新方法:
合并
使用何种操作清理工作树:
Stash
Shelve
如果当前分支的推送被拒，则自动更新
为提交和推送显示推送对话框
仅在提交到受保护分支时显示推送对话框
受保护分支:
master
使用凭据帮助程序
按路径筛选更新项目信息:
所有
确定
取消
应用(A)

(二)、创建仓库


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
GitMaven - pom.xml (GitMaven) - IntelliJ IDEA
GitMaven
Project
GitMaven F:\GitMaven
.idea
GitMaven.iml
pom.xml
src
External Libraries
Scratches and Consoles
Local History
Enable Version Control Integration...
VCS Operations Popup... Alt+`
Apply Patch...
Apply Patch from Clipboard...
Get from Version Control...
Import into Version Control
Browse VCS Repository
Sync Settings
Import into Subversion...
Share Project (Subversion)...
Create Git Repository...
Share Project on GitHub
Create Mercurial Repository
<artifactId>GitMa
<version>1.0-SNAP
</project>

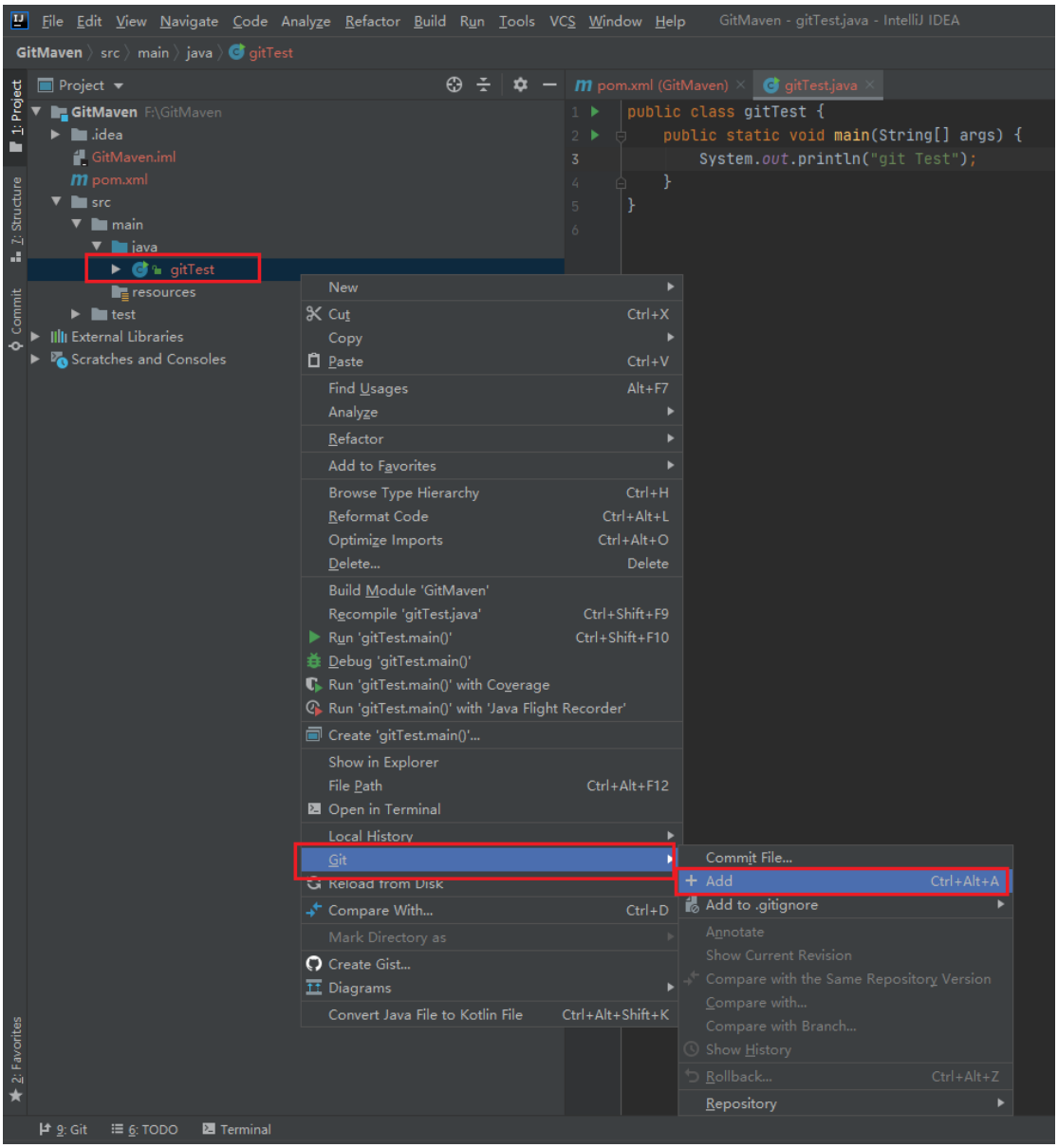


变红说明没有存在缓存区

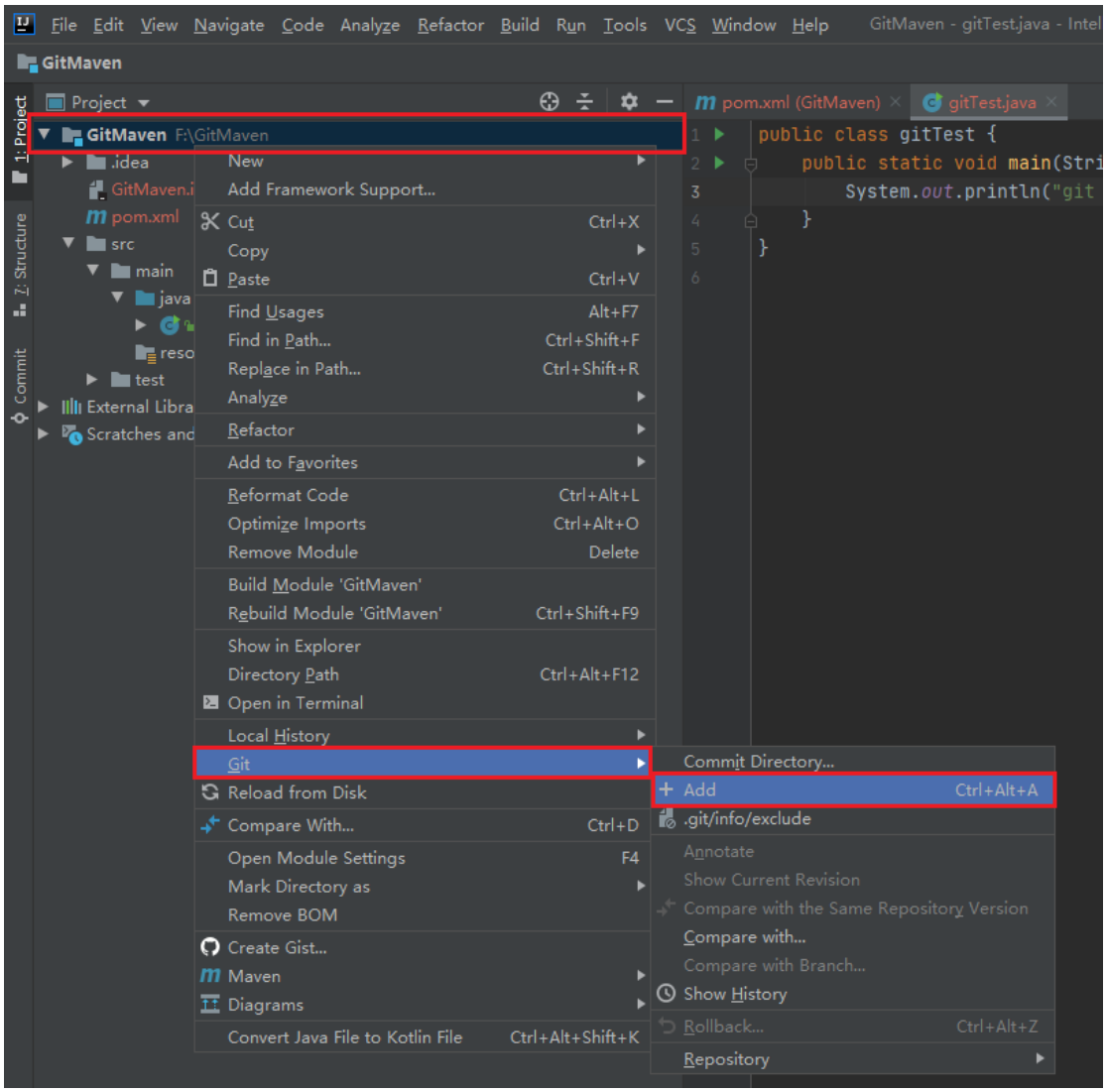


(三)、添加到缓存区

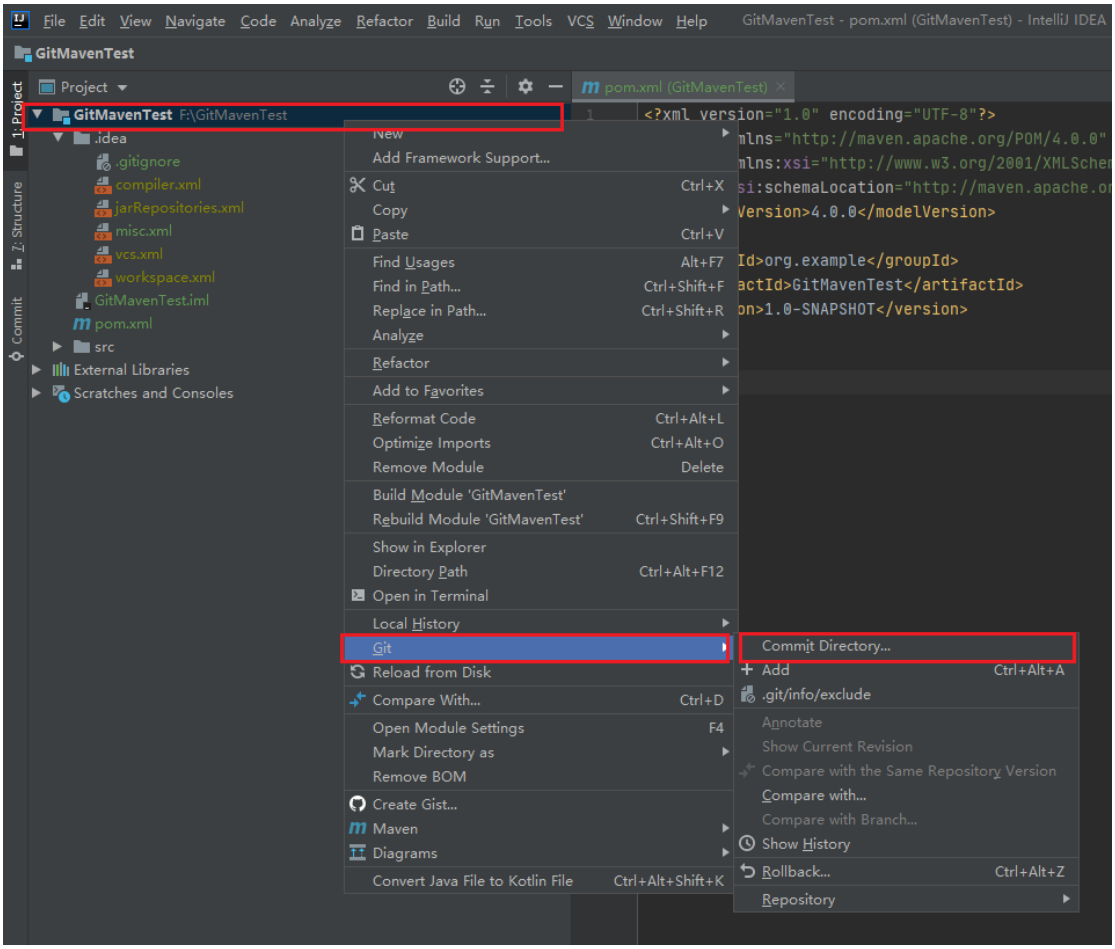
1、单个文件添加

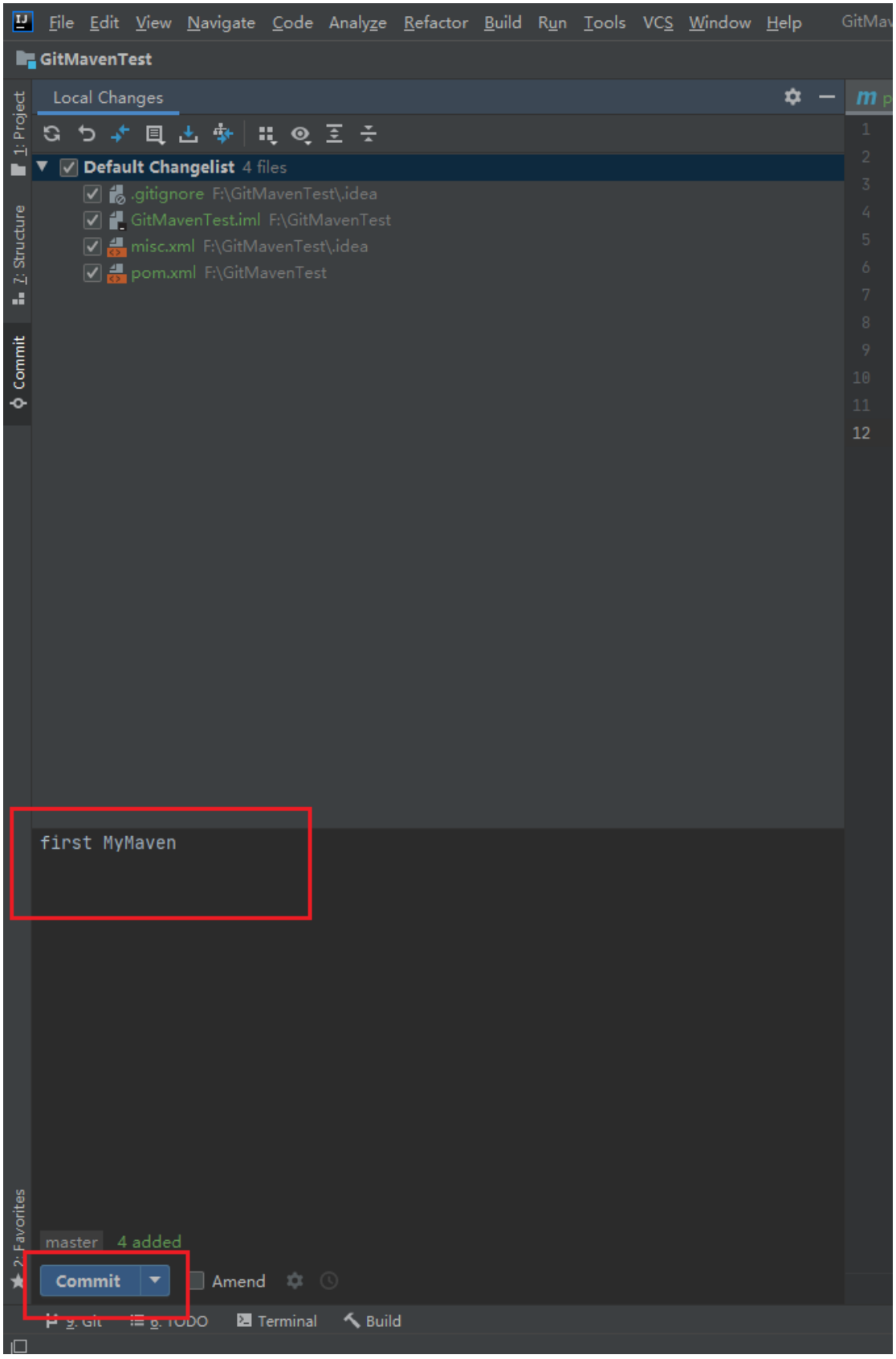


2、多个文件添加


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
GitMaven - gitTest.java - Intel
GitMaven
Project
GitMaven F:\GitMaven
.idea
GitMaven.i
pom.xml
src
main
java
reso
test
External Libra
Scratches and
pom.xml (GitMaven)
gitTest.java
public class gitTest {
public static void main(Stri
System.out.println("git
}
}
New
Add Framework Support...
Cut Ctrl+X
Copy
Paste Ctrl+V
Find Usages Alt+F7
Find in Path... Ctrl+Shift+F
Replace in Path... Ctrl+Shift+R
Analyze
Refactor
Add to Favorites
Reformat Code Ctrl+Alt+L
Optimize Imports Ctrl+Alt+O
Remove Module Delete
Build Module 'GitMaven'
Rebuild Module 'GitMaven' Ctrl+Shift+F9
Show in Explorer
Directory Path Ctrl+Alt+F12
Open in Terminal
Local History
Git
Reload from Disk
Compare With... Ctrl+D
Open Module Settings F4
Mark Directory as
Remove BOM
Create Gist...
Maven
Diagrams
Convert Java File to Kotlin File Ctrl+Alt+Shift+K
Commit Directory...
Add Ctrl+Alt+A
.git/info/exclude
Annotate
Show Current Revision
Compare with the Same Repository Version
Compare with...
Compare with Branch...
Show History
Rollback... Ctrl+Alt+Z
Repository

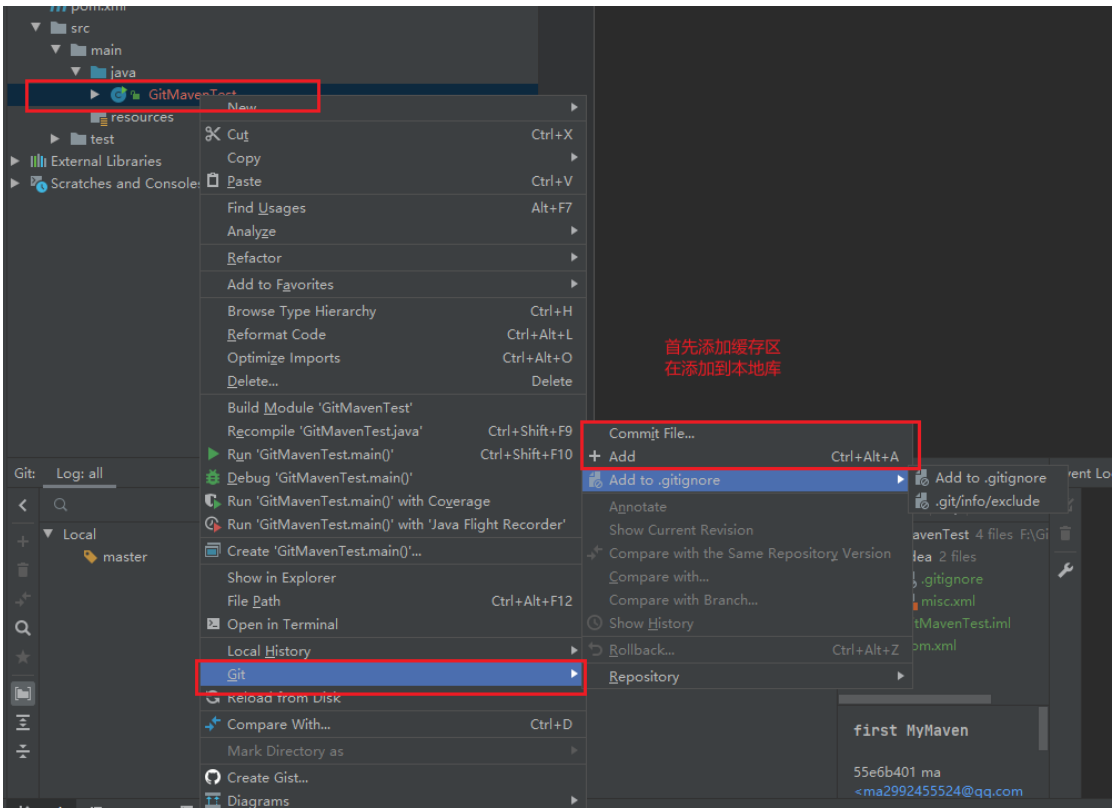
(四)、添加本地库


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
GitMavenTest - pom.xml (GitMavenTest) - IntelliJ IDEA
GitMavenTest
Project
pom.xml (GitMavenTest)
GitMavenTest F:\GitMavenTest
.idea
.gitignore
compiler.xml
jarRepositories.xml
misc.xml
vcs.xml
workspace.xml
GitMavenTest.iml
pom.xml
src
External Libraries
Scratches and Consoles
<?xml version="1.0" encoding="UTF-8"?>
New
Add Framework Support...
Cut Ctrl+X
Copy
Paste Ctrl+V
Find Usages Alt+F7
Find in Path... Ctrl+Shift+F
Replace in Path... Ctrl+Shift+R
Analyze
Refactor
Add to Favorites
Reformat Code Ctrl+Alt+L
Optimize Imports Ctrl+Alt+O
Remove Module Delete
Build Module 'GitMavenTest'
Rebuild Module 'GitMavenTest' Ctrl+Shift+F9
Show in Explorer
Directory Path Ctrl+Alt+F12
Open in Terminal
Local History
Git
Reload from Disk
Compare With... Ctrl+D
Open Module Settings F4
Mark Directory as
Remove BOM
Create Gist...
Maven
Diagrams
Convert Java File to Kotlin File Ctrl+Alt+Shift+K
Commit Directory...
Add Ctrl+Alt+A
.git/info/exclude
Annotate
Show Current Revision
Compare with the Same Repository Version
Compare with...
Compare with Branch...
Show History
Rollback... Ctrl+Alt+Z
Repository


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
GitMavenTest
Local Changes
Default Changelist 4 files
.gitignore F:\GitMavenTest\.idea
GitMavenTest.iml F:\GitMavenTest
misc.xml F:\GitMavenTest\.idea
pom.xml F:\GitMavenTest
first MyMaven
master 4 added
Commit
Amend
Terminal
Build
1: Project
7: Structure
Commit
2: Favorites

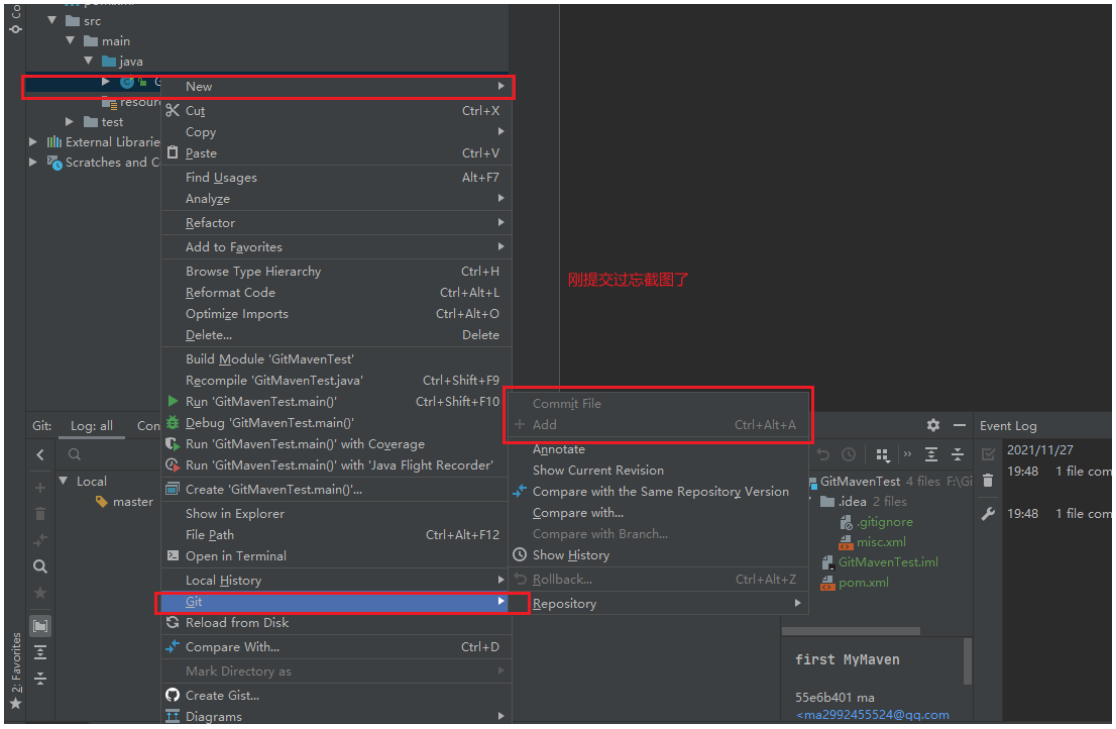
(五)、版本切换

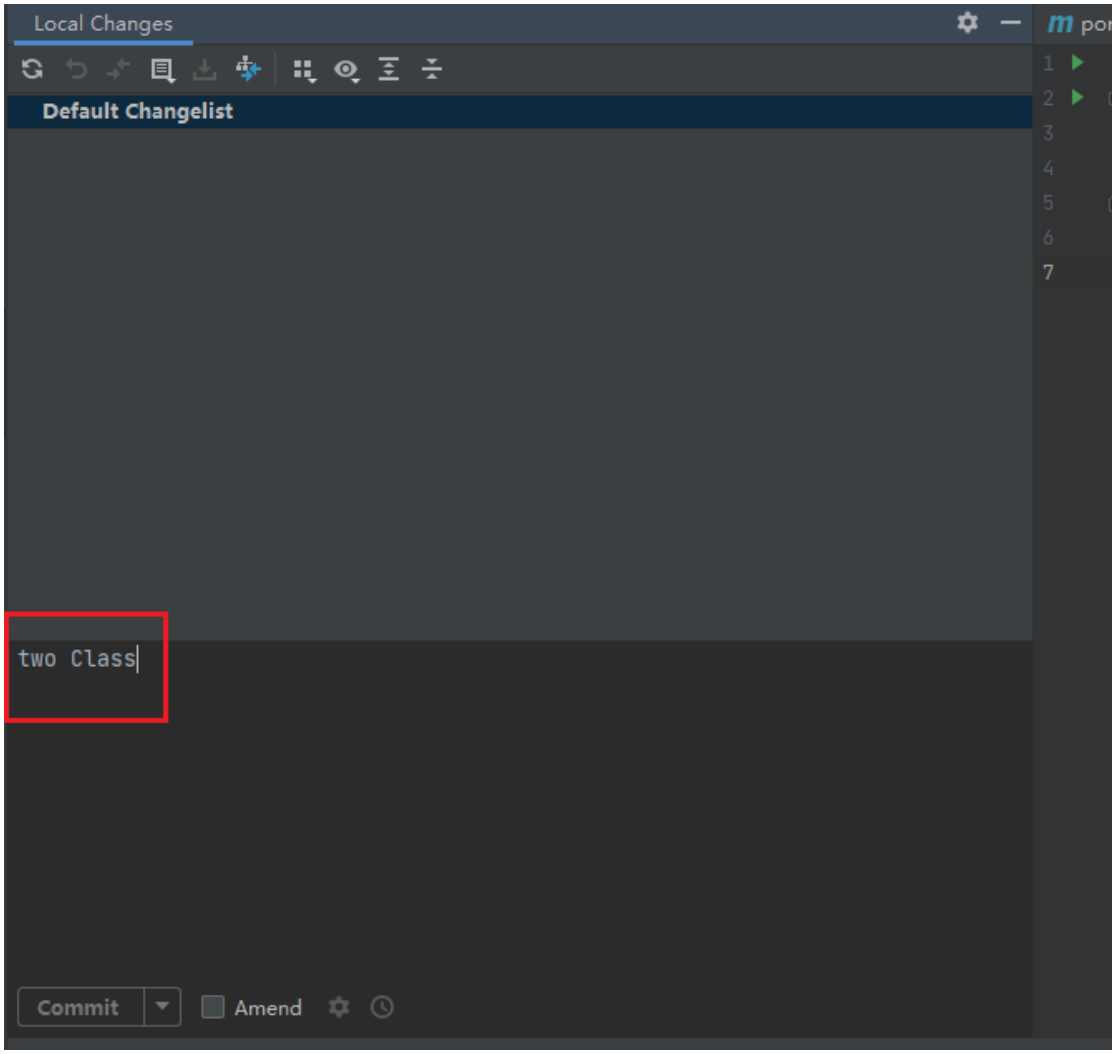
1、首先创建一个类提交到本地库



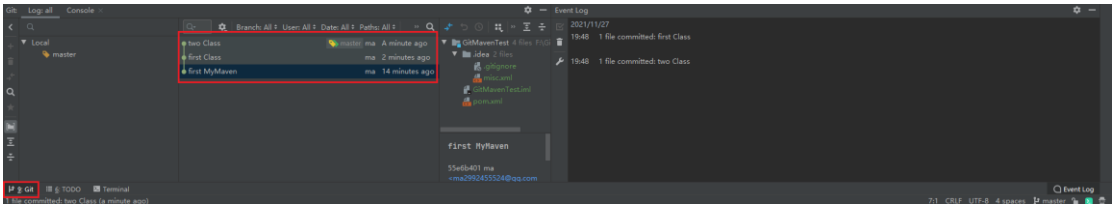
2、修改类再次提交

```
public class GitMavenTest {
    public static void main(String[] args) {
        System.out.println("git  maven test");
        System.out.println("git  maven test1");

    }
}
```

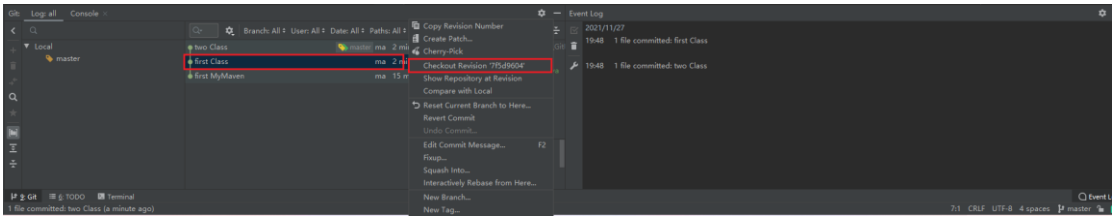

src
main
java
test
External Librarie
Scratches and C
New
Cut Ctrl+X
Copy
Paste Ctrl+V
Find Usages Alt+F7
Analyze
Refactor
Add to Favorites
Browse Type Hierarchy Ctrl+H
Reformat Code Ctrl+Alt+L
Optimize Imports Ctrl+Alt+O
Delete... Delete
Build Module 'GitMavenTest'
Recompile 'GitMavenTest.java' Ctrl+Shift+F9
Run 'GitMavenTest.main()' Ctrl+Shift+F10
Debug 'GitMavenTest.main()'
Run 'GitMavenTest.main()' with Coverage
Run 'GitMavenTest.main()' with 'Java Flight Recorder'
Create 'GitMavenTest.main()'...
Show in Explorer
File Path Ctrl+Alt+F12
Open in Terminal
Local History
Git
Reload from Disk
Compare With... Ctrl+D
Mark Directory as
Create Gist...
Diagrams
刚提交过忘截图了
Commit File
Add Ctrl+Alt+A
Annotate
Show Current Revision
Compare with the Same Repository Version
Compare with...
Compare with Branch...
Show History
Rollback... Ctrl+Alt+Z
Repository
Git: Log: all
Local
master
Event Log
2021/11/27
19:48 1 file com
GitMavenTest 4 files
.idea 2 files
.gitignore
misc.xml
GitMavenTest.iml
pom.xml
first MyMaven
55e6b401 ma
<ma2992455524@qq.com
2: Favorites


Local Changes
Default Changelist
two Class
Commit
Amend

3、查看全部版本



4、切换版本

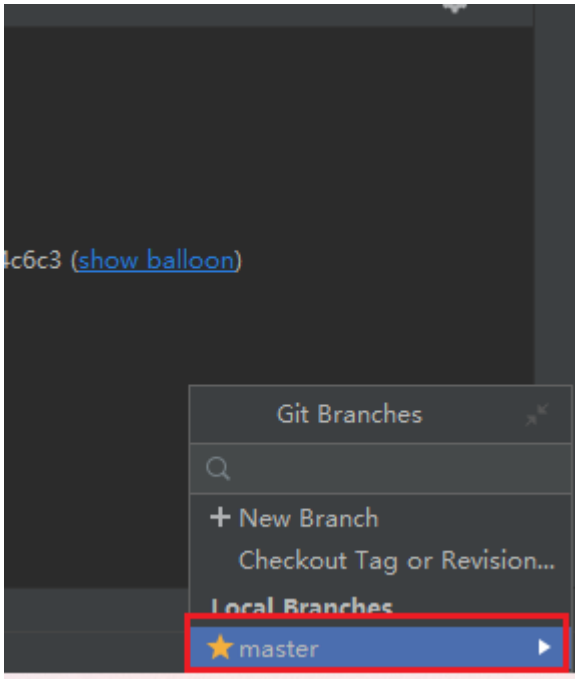


5、查看这个类

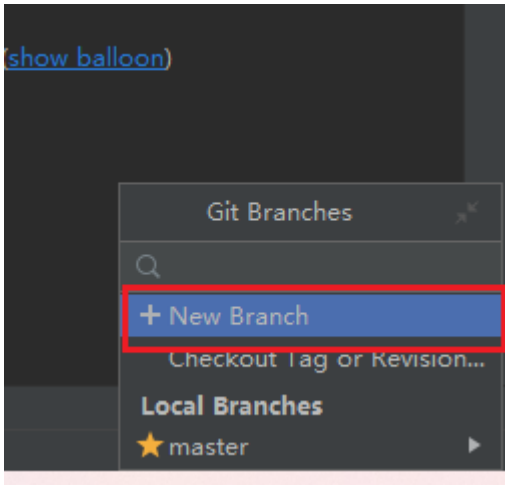
```
public class GitMavenTest {
    public static void main(String[] args) {
        System.out.println("git  maven test");
    }
}
```

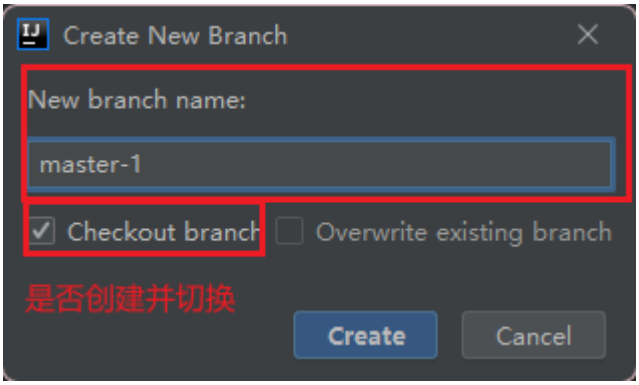
(六)、创建分支

1、查看当前分支

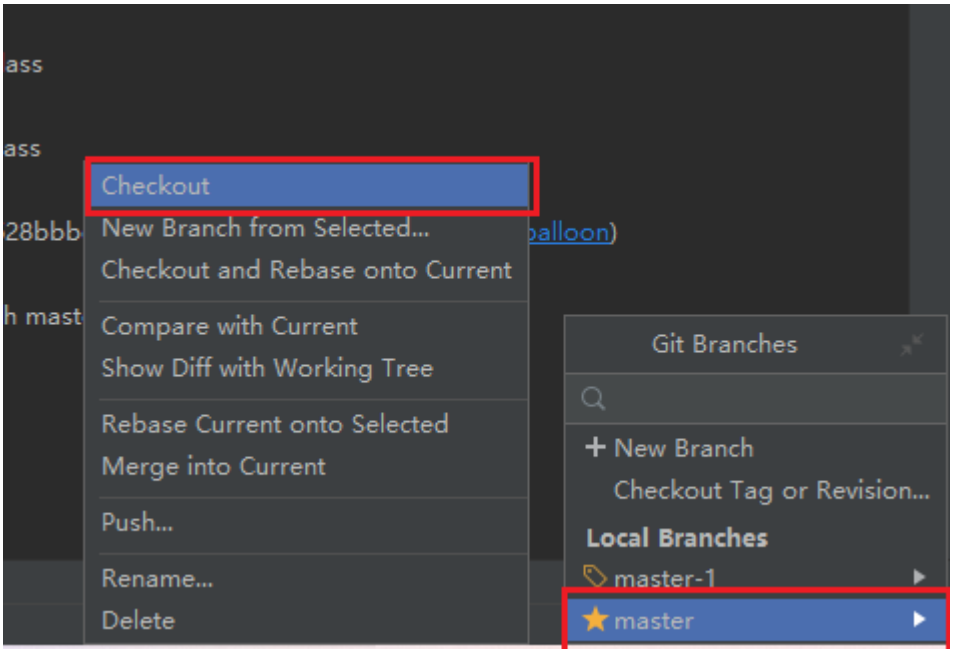


2、创建分支




Create New Branch
New branch name:
master-1
Checkout branch
Overwrite existing branch
是否创建并切换
Create
Cancel

3、切换分支

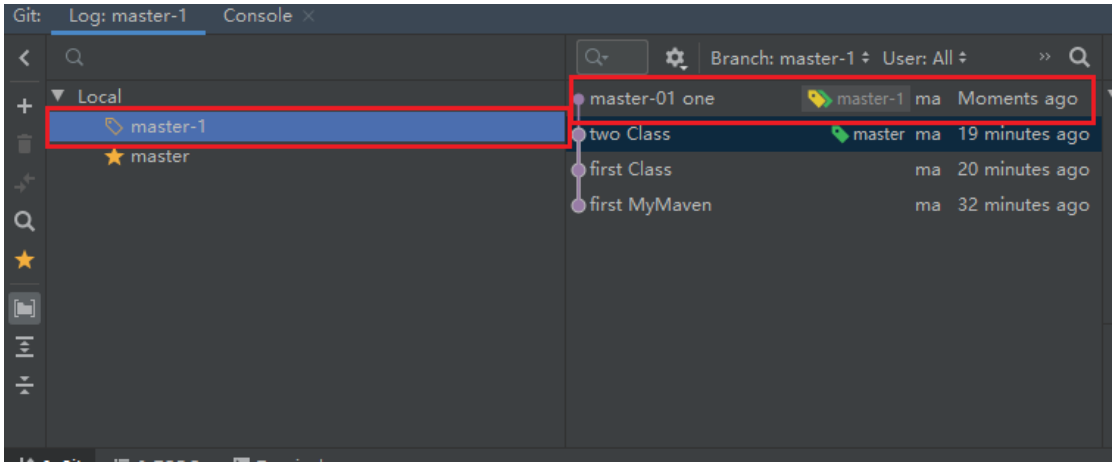

Checkout
New Branch from Selected...
Checkout and Rebase onto Current
Compare with Current
Show Diff with Working Tree
Rebase Current onto Selected
Merge into Current
Push...
Rename...
Delete
Git Branches
New Branch
Checkout Tag or Revision...
Local Branches
master-1
master

(七)、合并分支(正常合并)

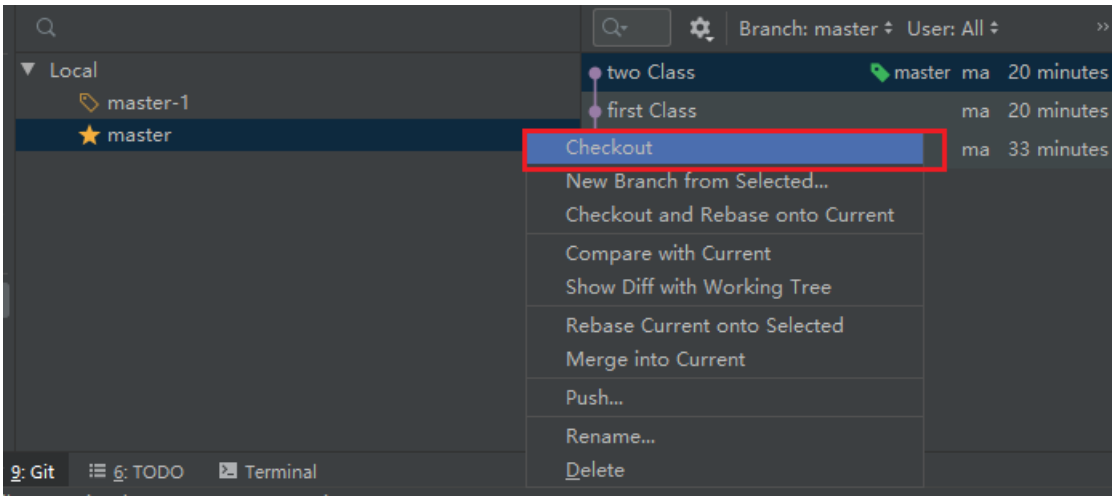
1、切换其他分支添加内容,添加了 2 和 3

```
public class GitMavenTest {
    public static void main(String[] args) {
        System.out.println("git  maven test");
        System.out.println("git  maven test1");
        System.out.println("git  maven test2");
        System.out.println("git  maven test3");
    }
}
```

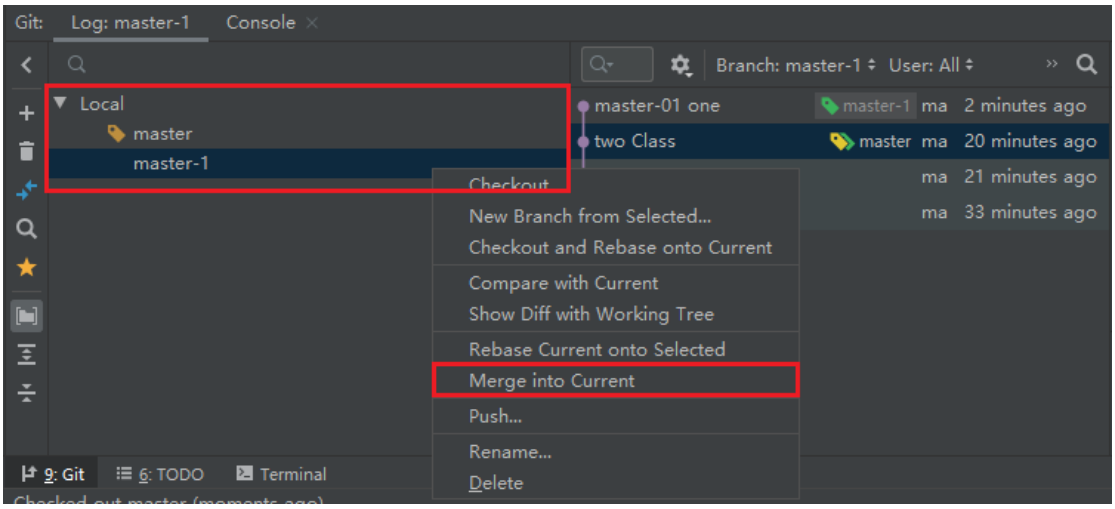
2、存储缓存区,提交本地库



3、切换到 master 分支



4、合并 master-1 分支



5、master 的类信息变了

```java
public class GitMavenTest {
    public static void main(String[] args) {
        System.out.println("git  maven test");
        System.out.println("git  maven test1");
        System.out.println("git  maven test2");
        System.out.println("git  maven test3");
    }
}
```

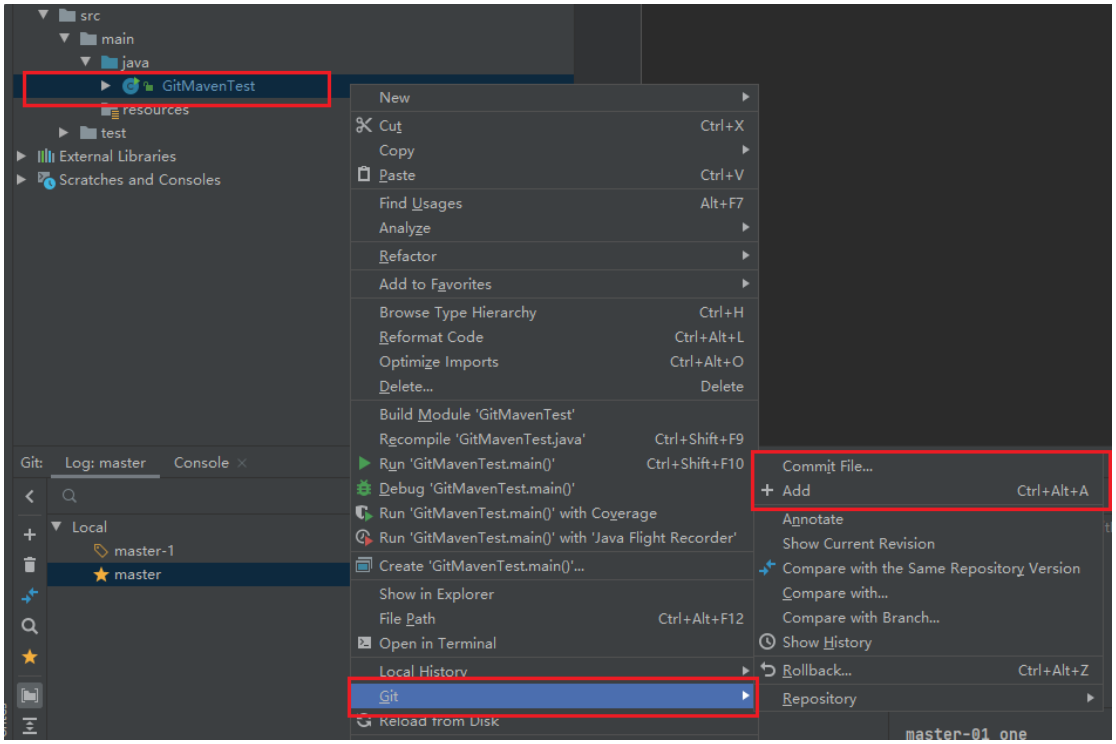
(八)、合并分支(冲突合并)

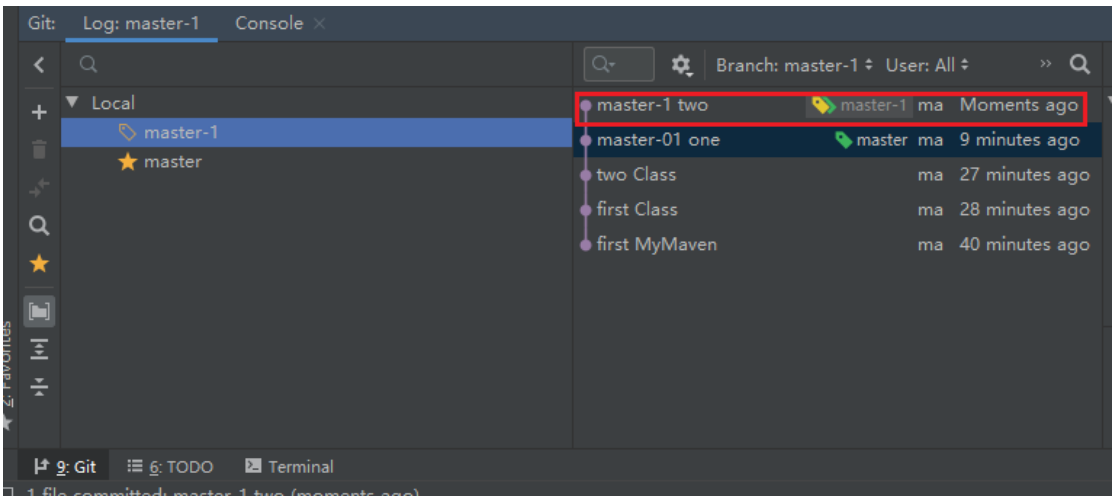
1、首先来到 master-1 分支修改类信息

多添加一行 master-1 test

```
public class GitMavenTest {
    public static void main(String[] args) {
        System.out.println("git  maven test");
        System.out.println("git  maven test1");
        System.out.println("git  maven test2");
        System.out.println("git  maven test3");
        System.out.println("master-1 Test");
    }
}
```

2、master-1 提交暂存区和本地库

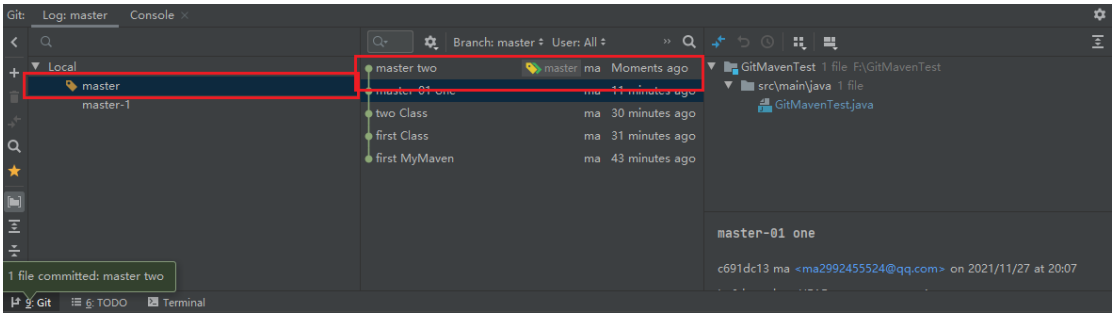




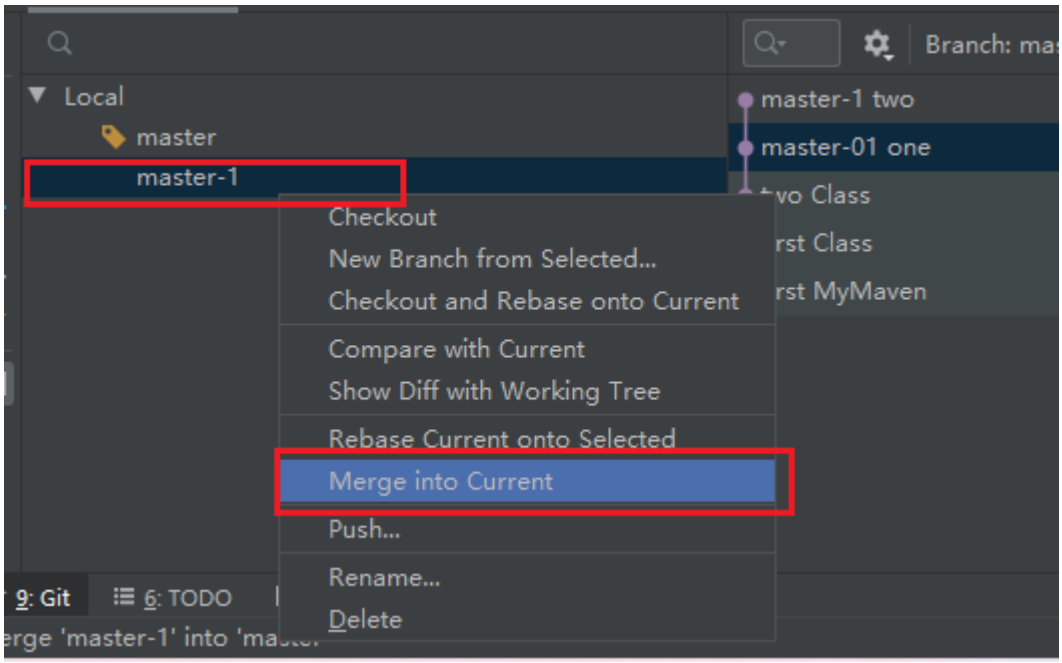
3、切换回 master 添加类信息

```java
public class GitMavenTest {
    public static void main(String[] args) {
        System.out.println("git  maven test");
        System.out.println("git  maven test1");
        System.out.println("git  maven test2");
        System.out.println("git  maven test3");
        System.out.println("master  test");
    }
}
```

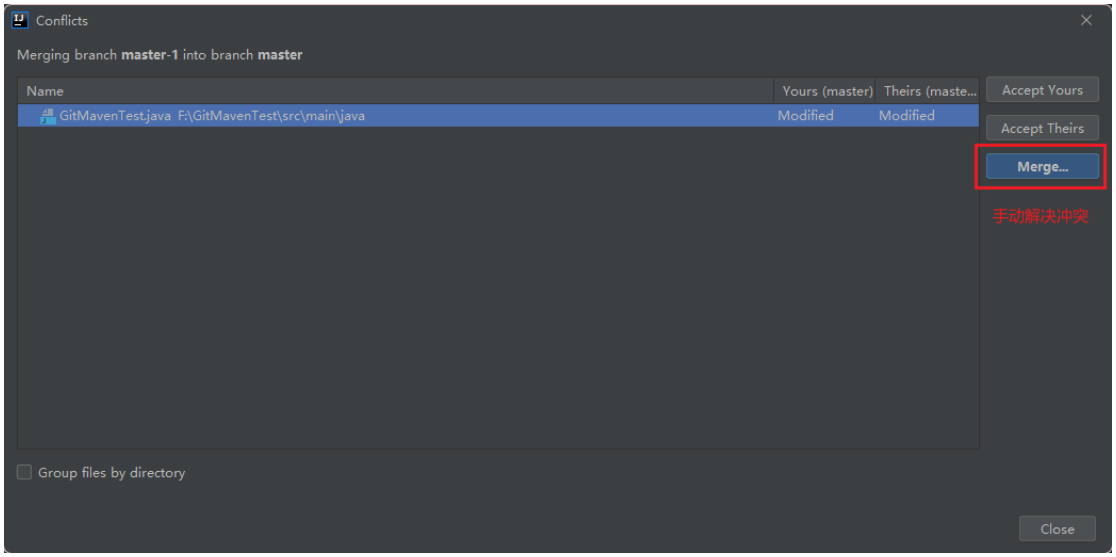
4、提交到缓存区和本地库



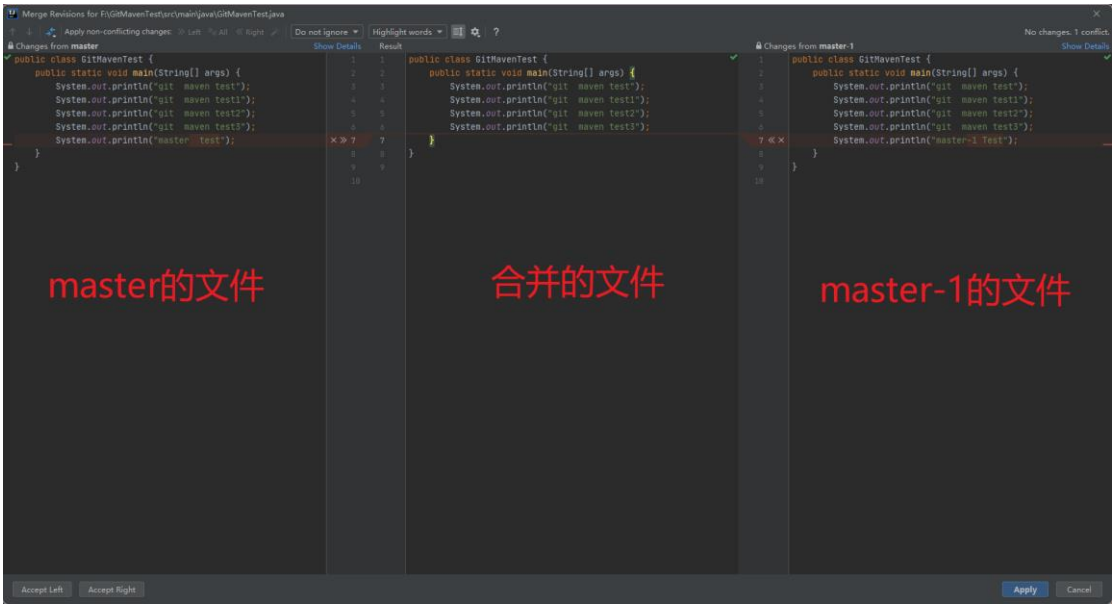
5、双方都添加了一行产生冲突,开始合并



6、弹出冲突提示

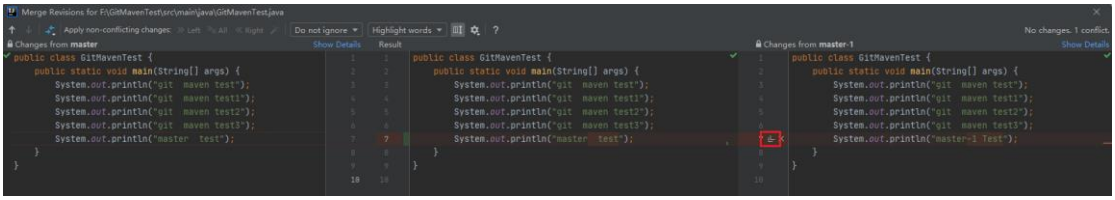


7、冲突解决界面

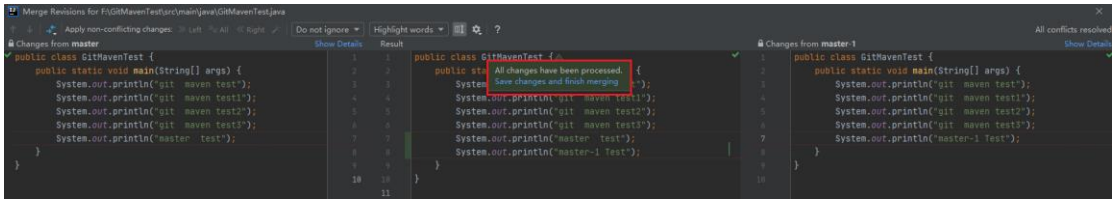


8、将 master 的添加到合并文件中

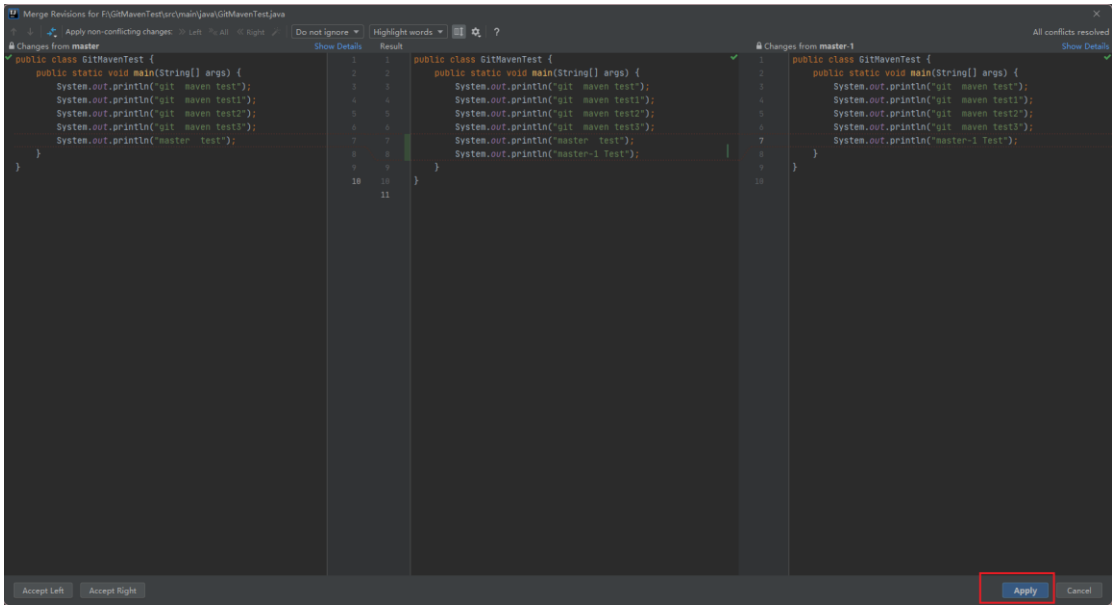
9、将 master-1 的添加到 master 的下面



10、提示可以提交



11、应用



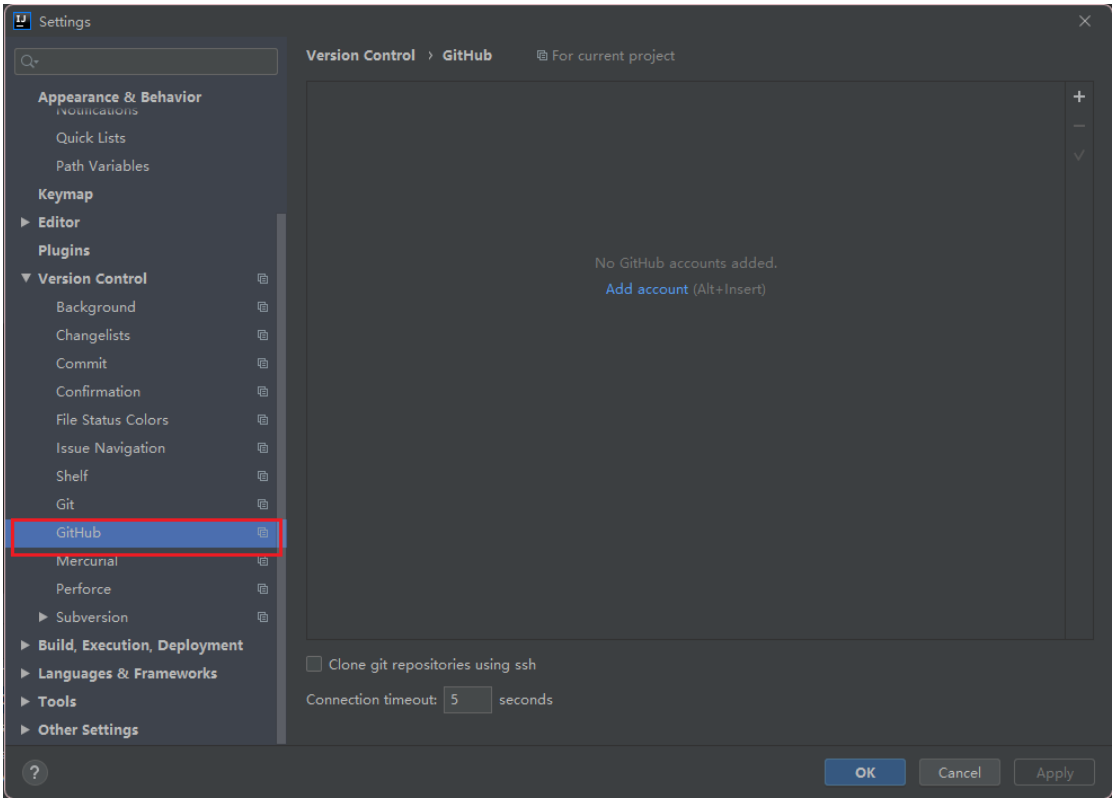
12、查看文件

```
public class GitMavenTest {
    public static void main(String[] args) {
        System.out.println("git  maven test");
        System.out.println("git  maven test1");
        System.out.println("git  maven test2");
        System.out.println("git  maven test3");
        System.out.println("master  test");
        System.out.println("master-1 Test");
    }
}
```

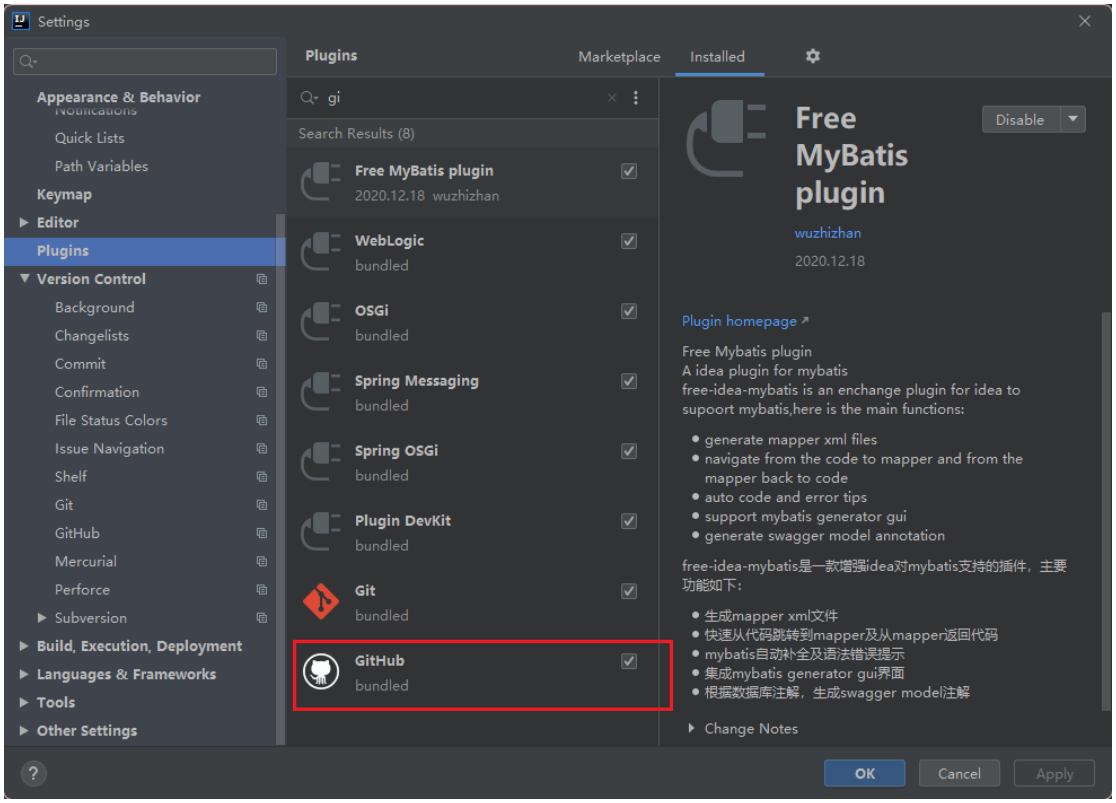
五、IDEA 使用 GitHub

(一)、IDEA 编译器登录 GitHub

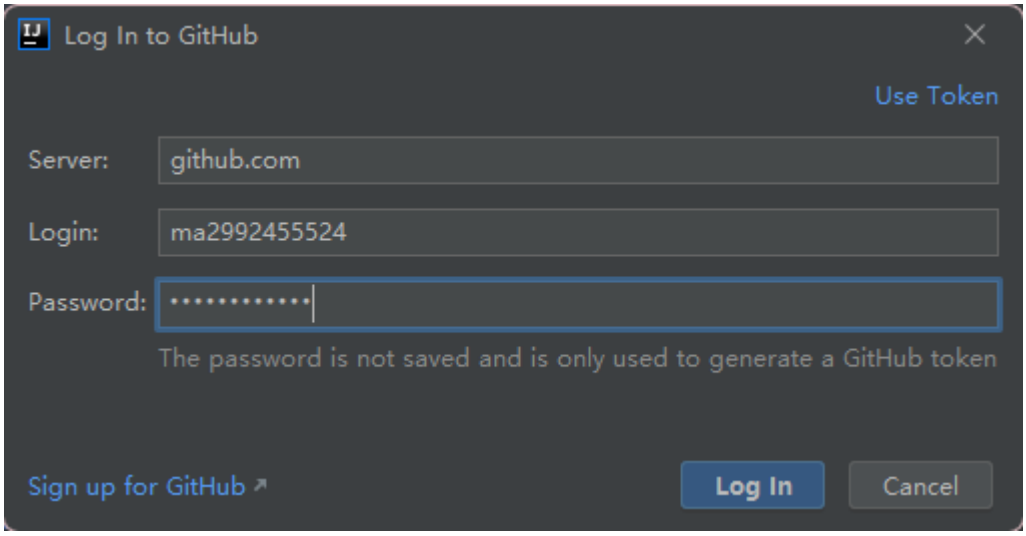
1、打开设置查看有没有 github



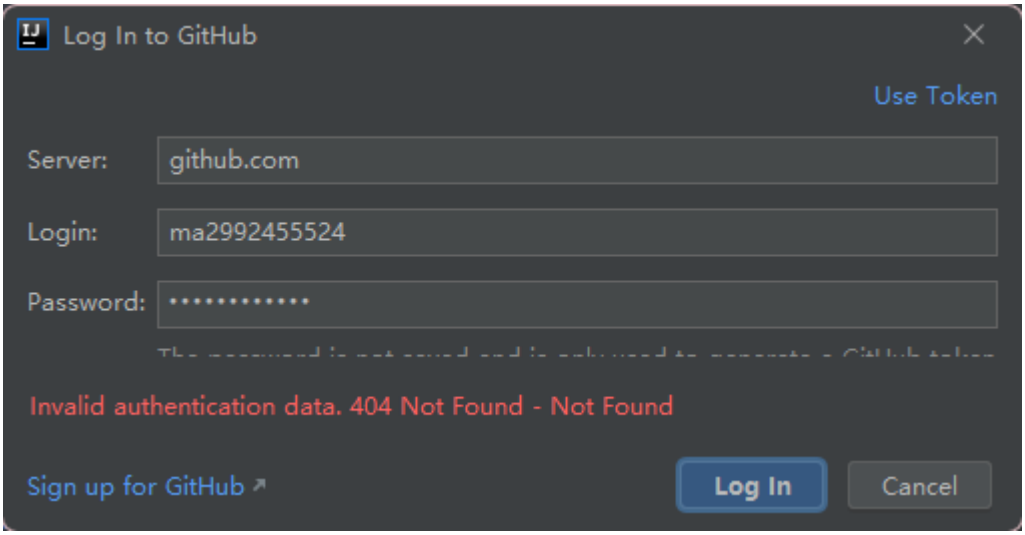
2、如果没有的话去插件仓库下载



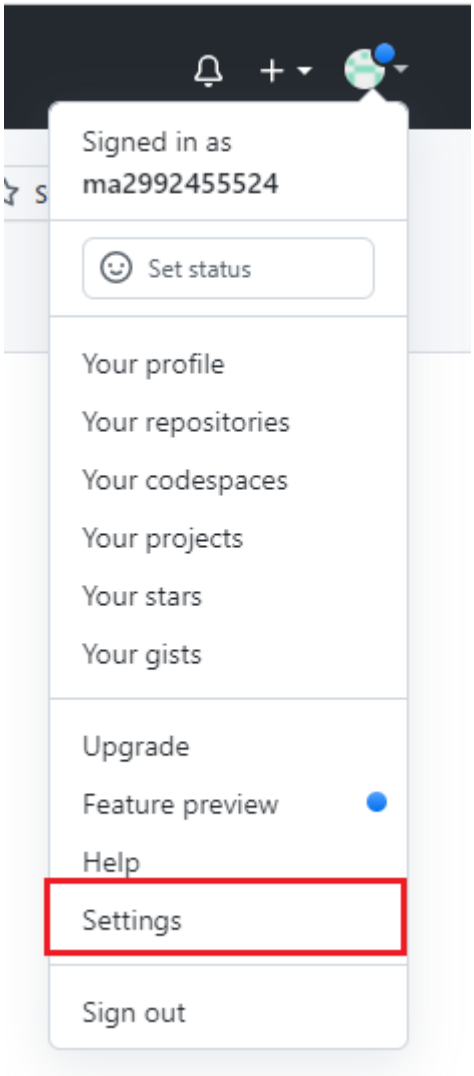
3、登录账户

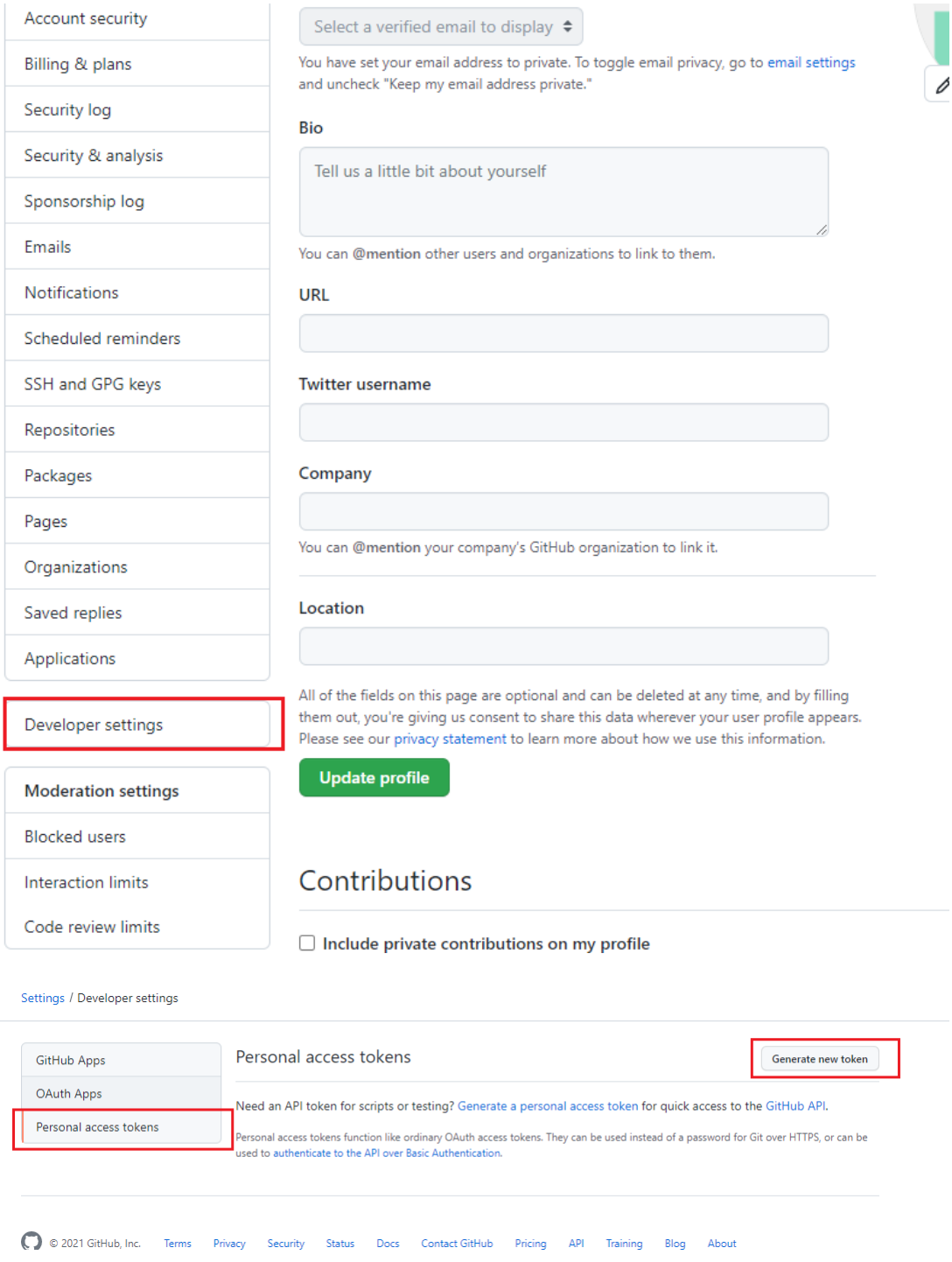


4、很难登上,使用 token 登录

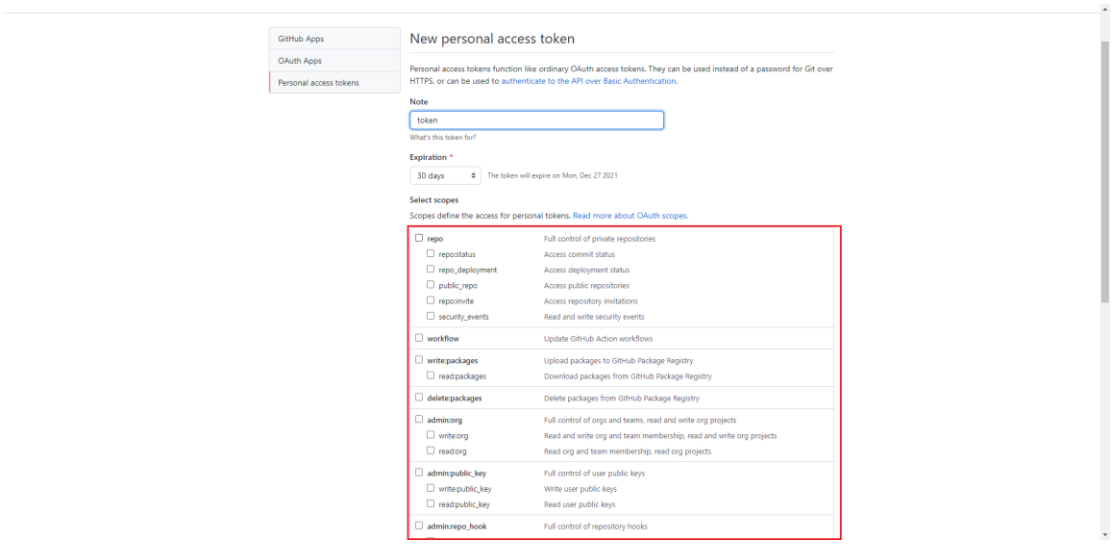


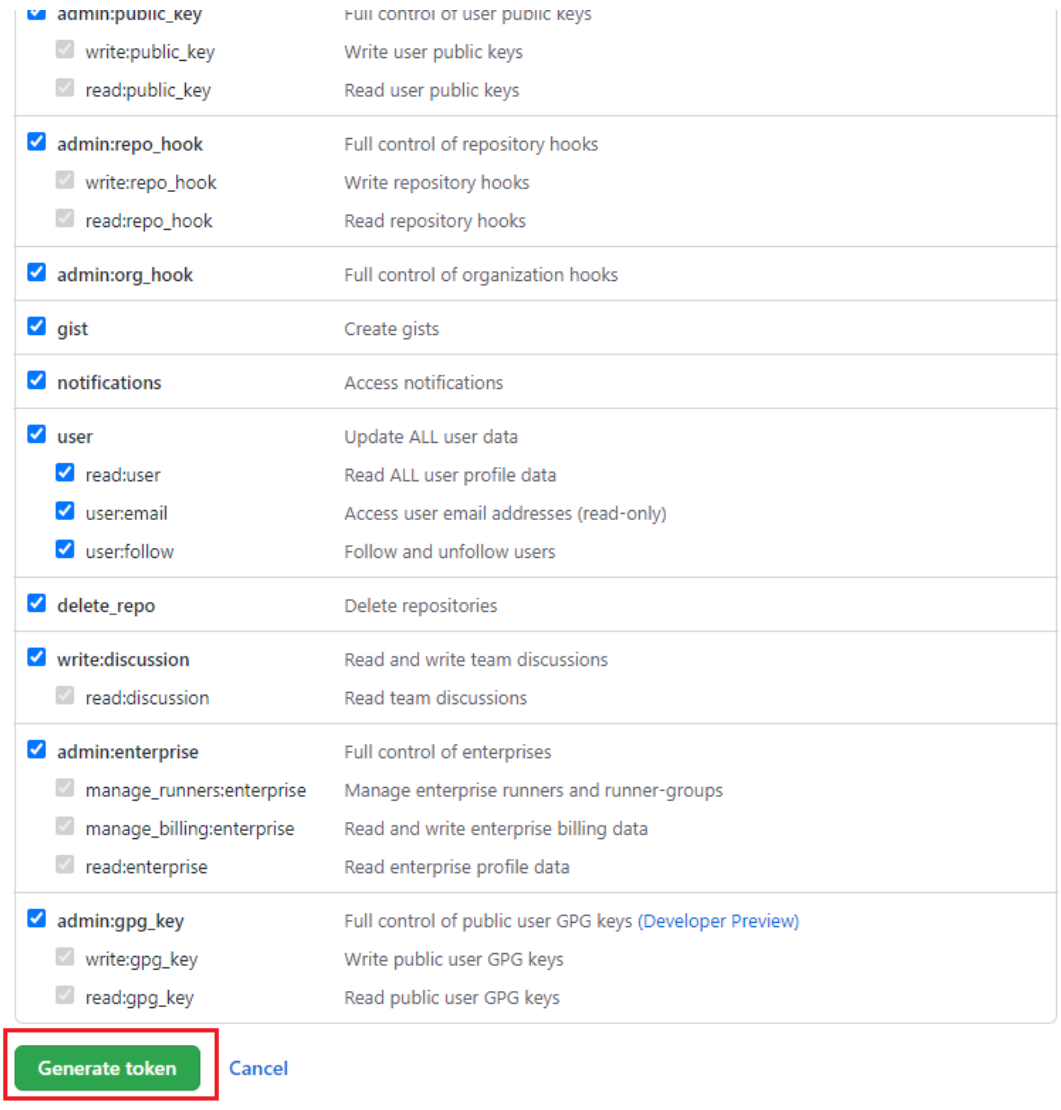
5、创建一个 token


Signed in as
ma2992455524
Set status
Your profile
Your repositories
Your codespaces
Your projects
Your stars
Your gists
Upgrade
Feature preview
Help
Settings
Sign out

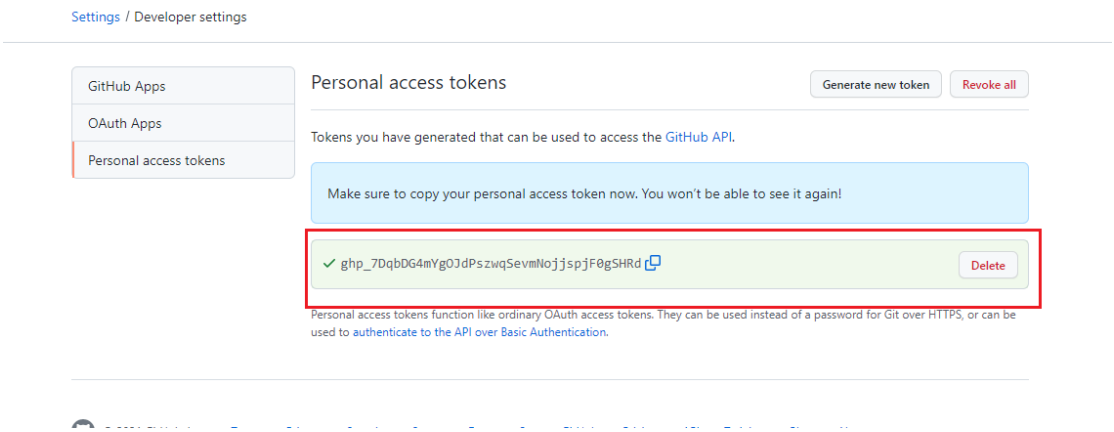

Account security
Billing & plans
Security log
Security & analysis
Sponsorship log
Emails
Notifications
Scheduled reminders
SSH and GPG keys
Repositories
Packages
Pages
Organizations
Saved replies
Applications
Developer settings
Moderation settings
Blocked users
Interaction limits
Code review limits
Select a verified email to display
You have set your email address to private. To toggle email privacy, go to email settings and uncheck "Keep my email address private."
Bio
Tell us a little bit about yourself
You can @mention other users and organizations to link to them.
URL
Twitter username
Company
You can @mention your company's GitHub organization to link it.
Location
All of the fields on this page are optional and can be deleted at any time, and by filling them out, you're giving us consent to share this data wherever your user profile appears. Please see our privacy statement to learn more about how we use this information.
Update profile
Contributions
Include private contributions on my profile

Settings / Developer settings
GitHub Apps
OAuth Apps
Personal access tokens
Personal access tokens
Generate new token
Need an API token for scripts or testing? Generate a personal access token for quick access to the GitHub API.
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.
© 2021 GitHub, Inc. Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

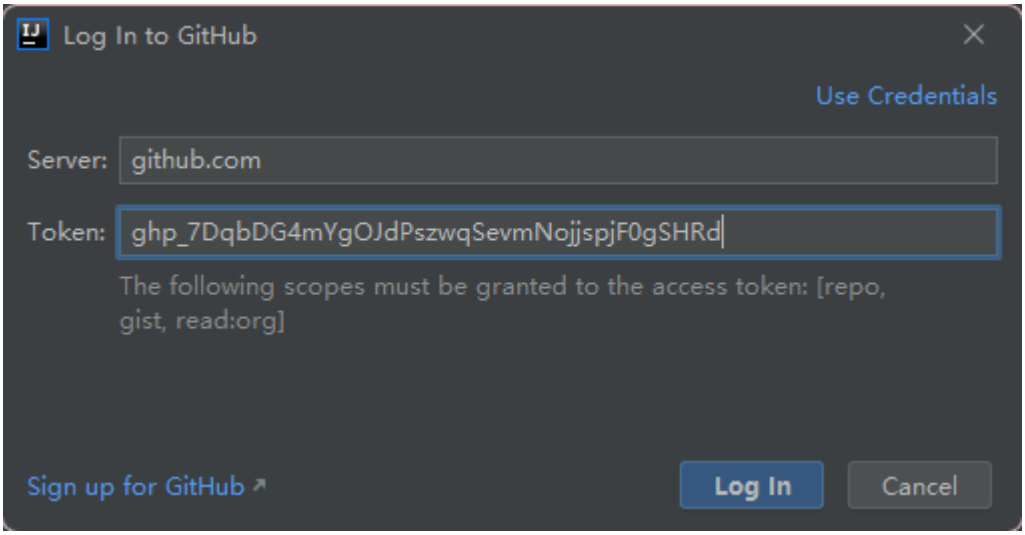




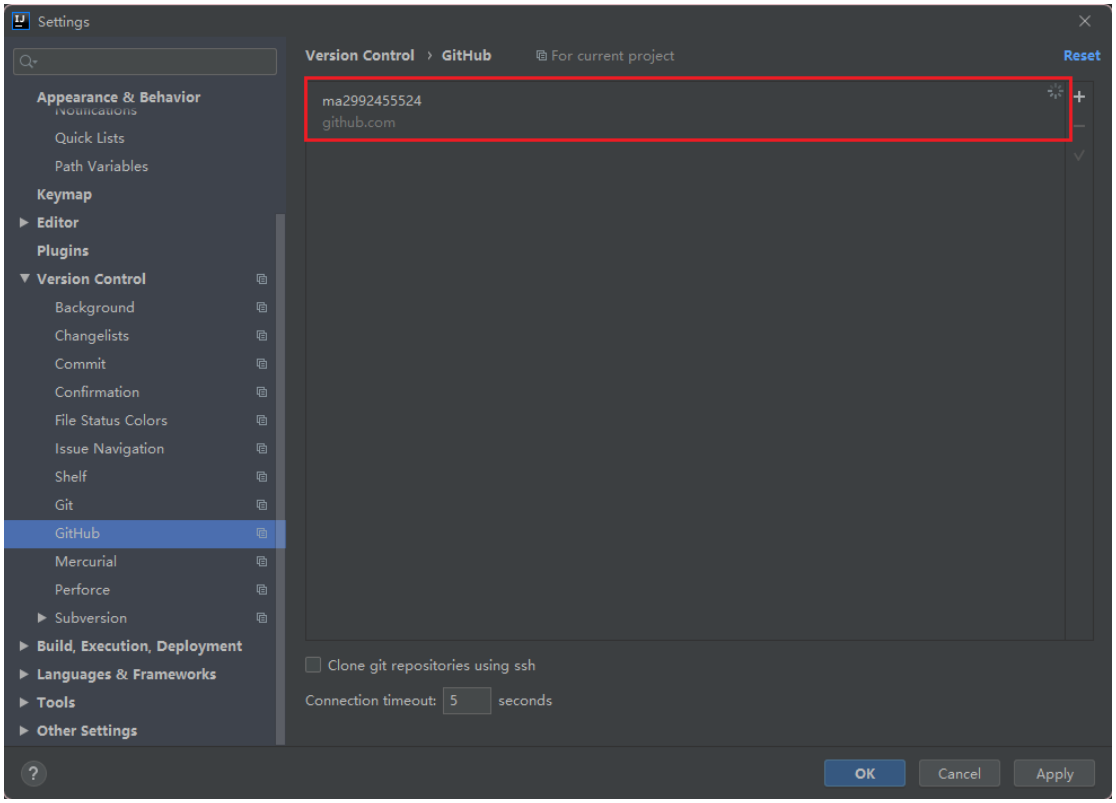
6、复制 token,注意一刷新就看不见了



7、IDEA 使用 token 登录

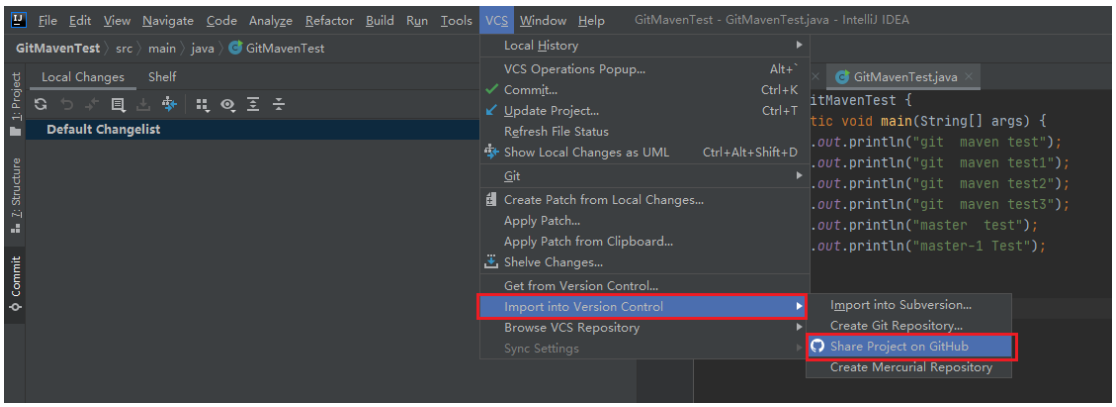


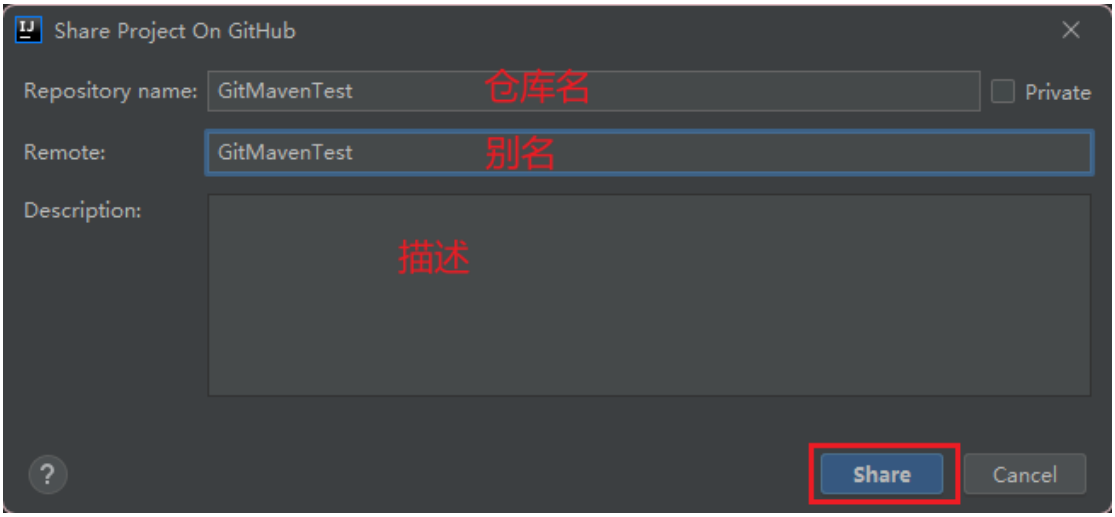
8、登录成功



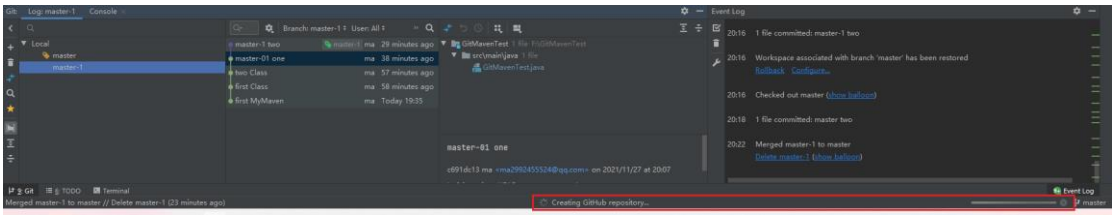
(二)、使用 IDEA 创建远程库并上传

1、创建并上传

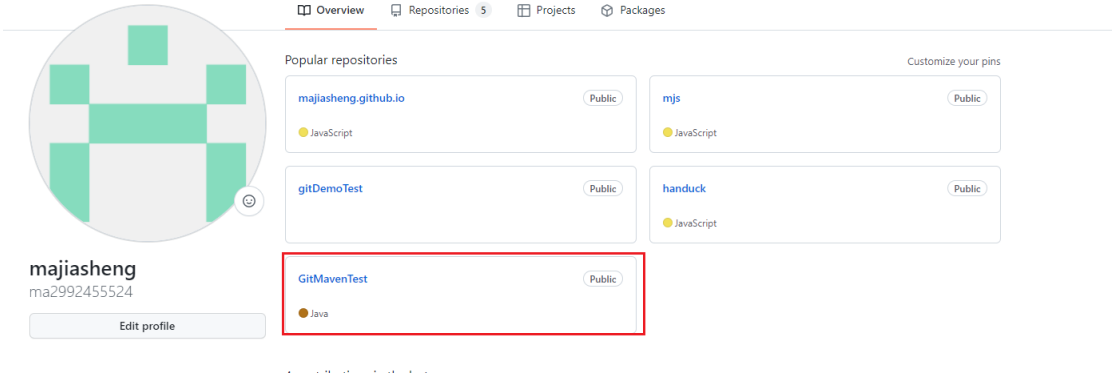


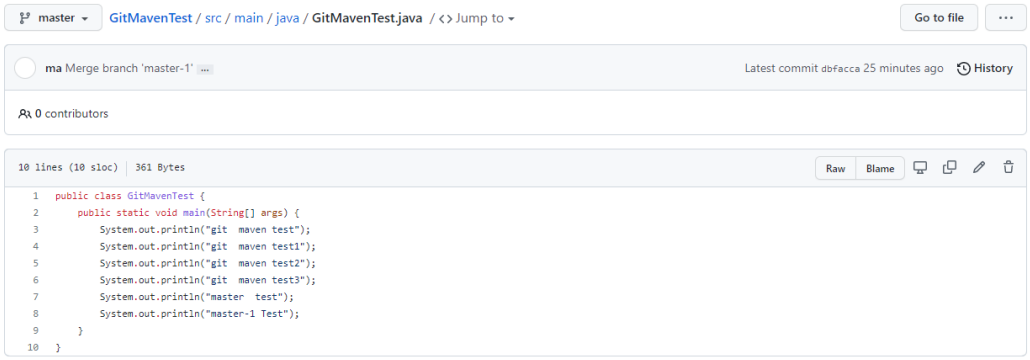

Share Project On GitHub
Repository name:
GitMavenTest
仓库名
Private
Remote:
GitMavenTest
别名
Description:
描述
Share
Cancel

2、创建中



3、验证


Overview
Repositories 5
Projects
Packages
Popular repositories
Customize your pins
majiasheng.github.io
Public
JavaScript
mjs
Public
JavaScript
gitDemoTest
Public
handuck
Public
JavaScript
GitMavenTest
Public
Java
majiasheng
ma2992455524
Edit profile

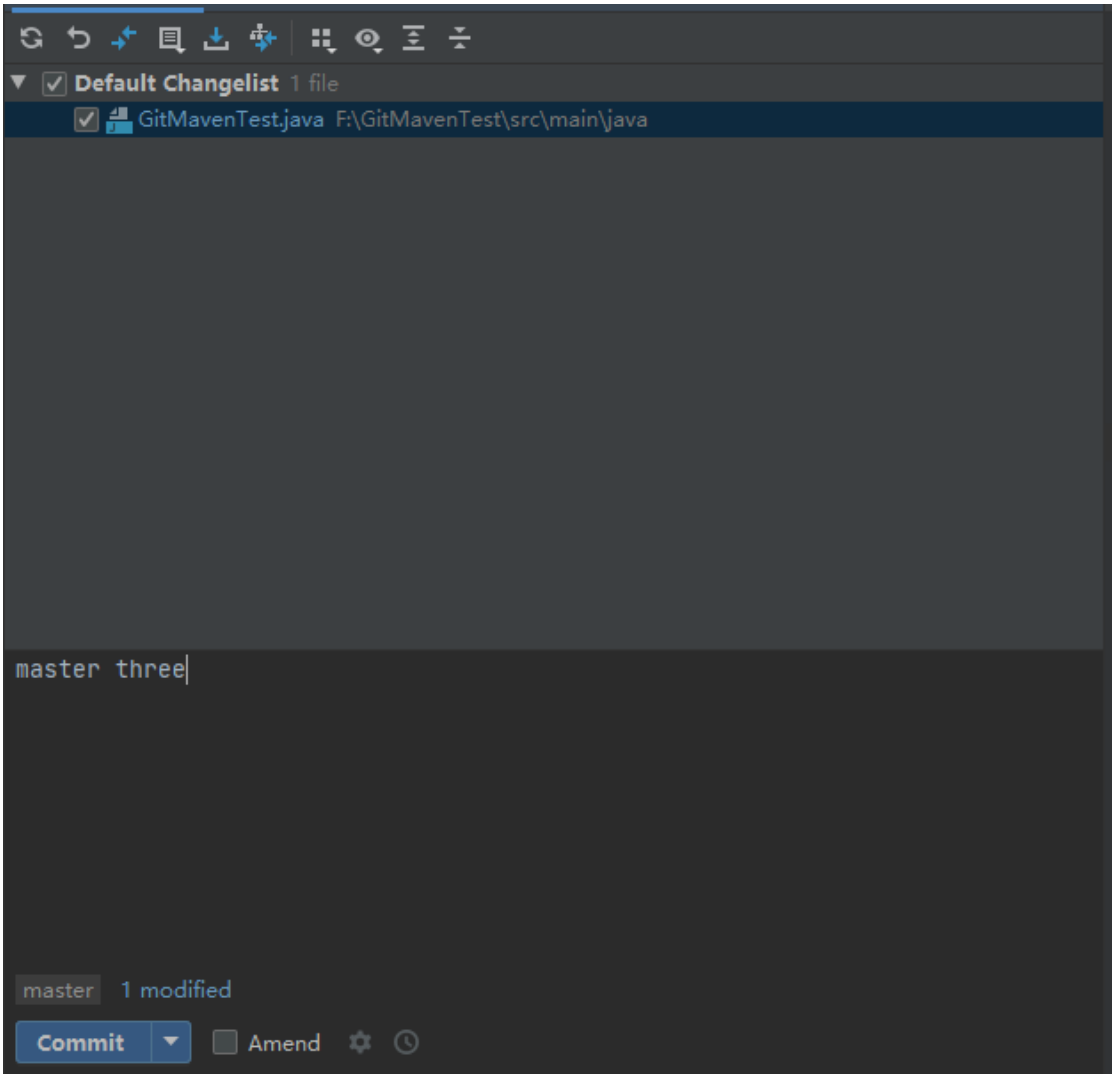


(三)、上传到远程库

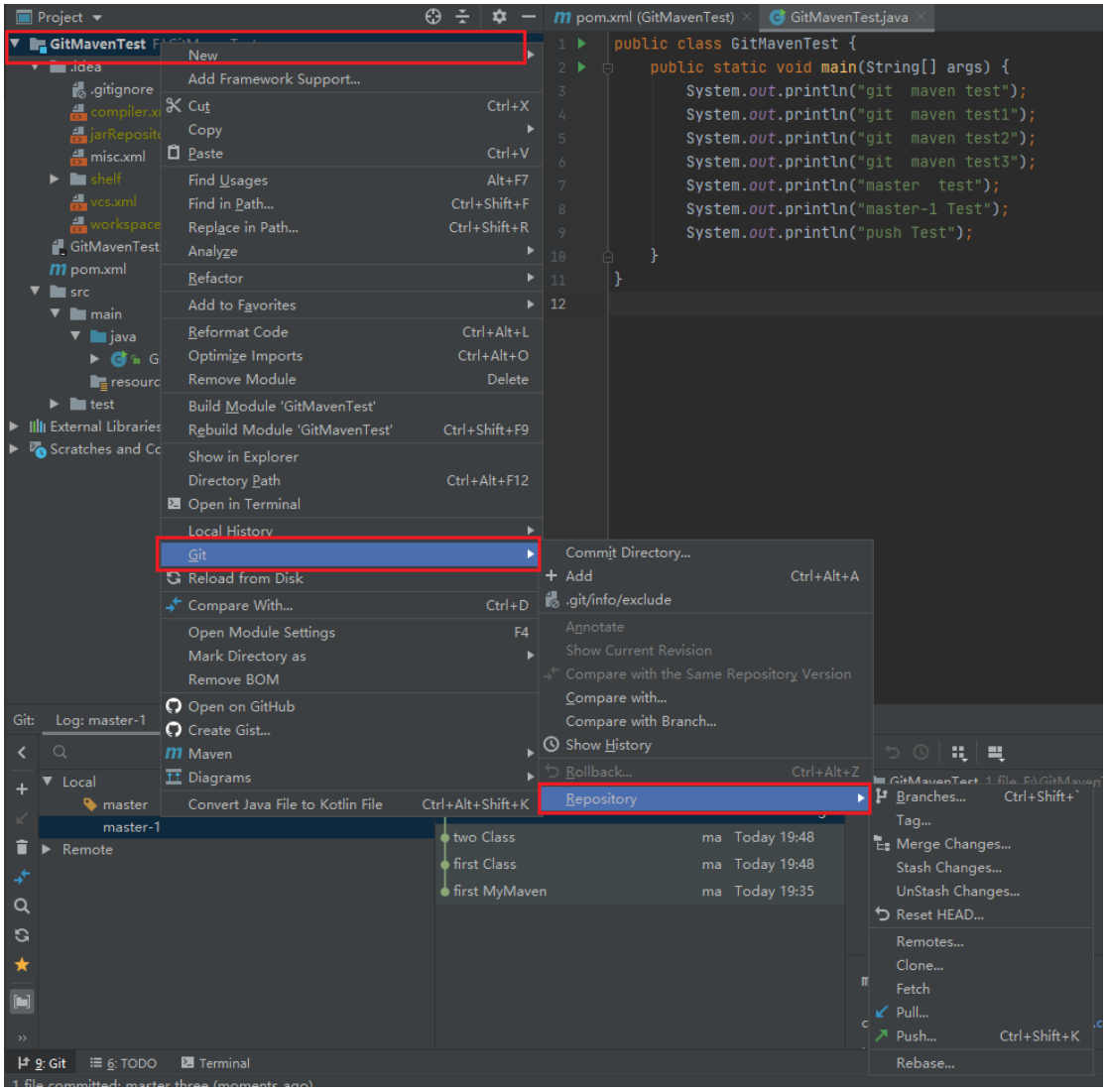
1、修改现在的类信息

```java
public class GitMavenTest {
    public static void main(String[] args) {
        System.out.println("git  maven test");
        System.out.println("git  maven test1");
        System.out.println("git  maven test2");
        System.out.println("git  maven test3");
        System.out.println("master  test");
        System.out.println("master-1 Test");
        System.out.println("push Test");
    }
}
```

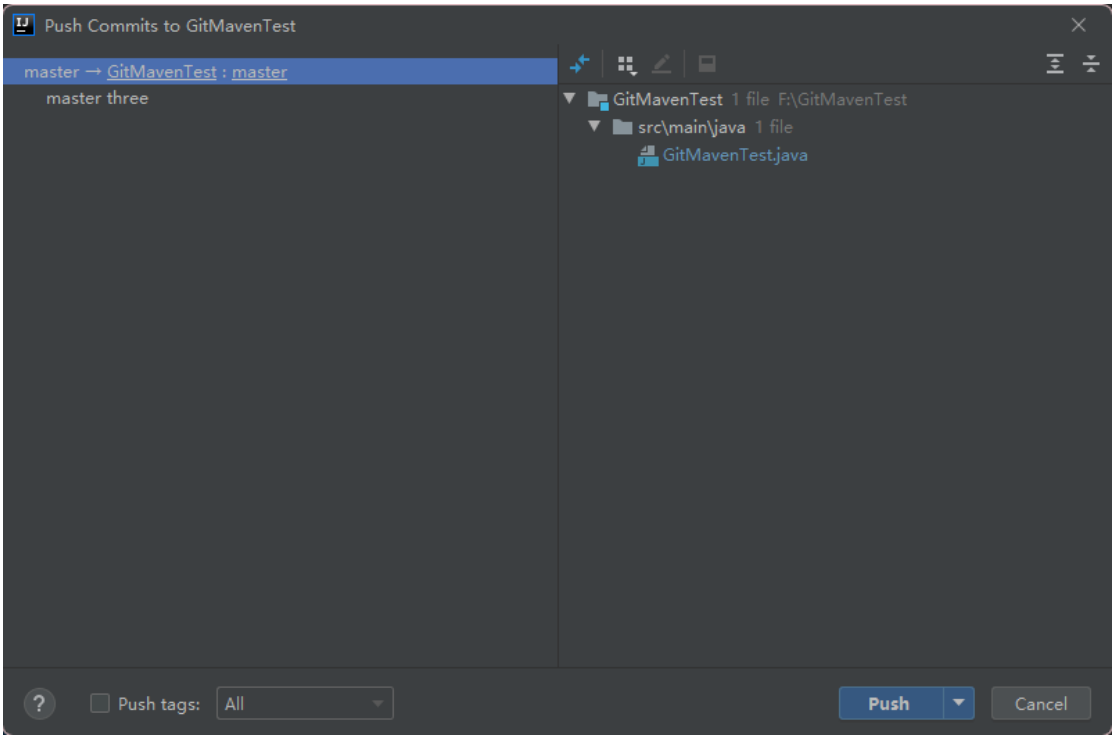
2、添加暂存区,本地仓库


Default Changelist 1 file
GitMavenTest.java F:\GitMavenTest\src\main\java
master three
master 1 modified
Commit
Amend

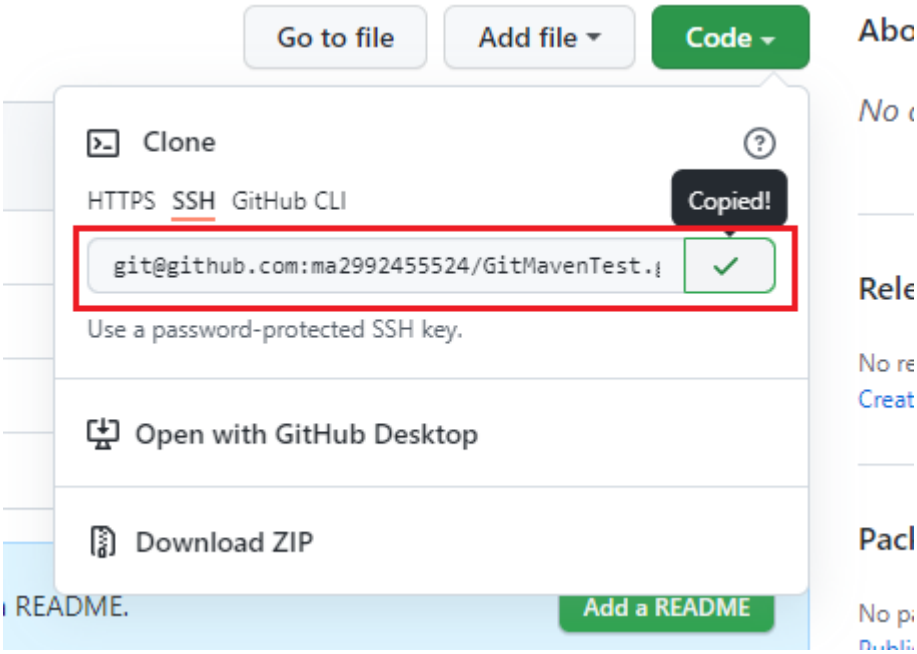
4、上传到远程仓库

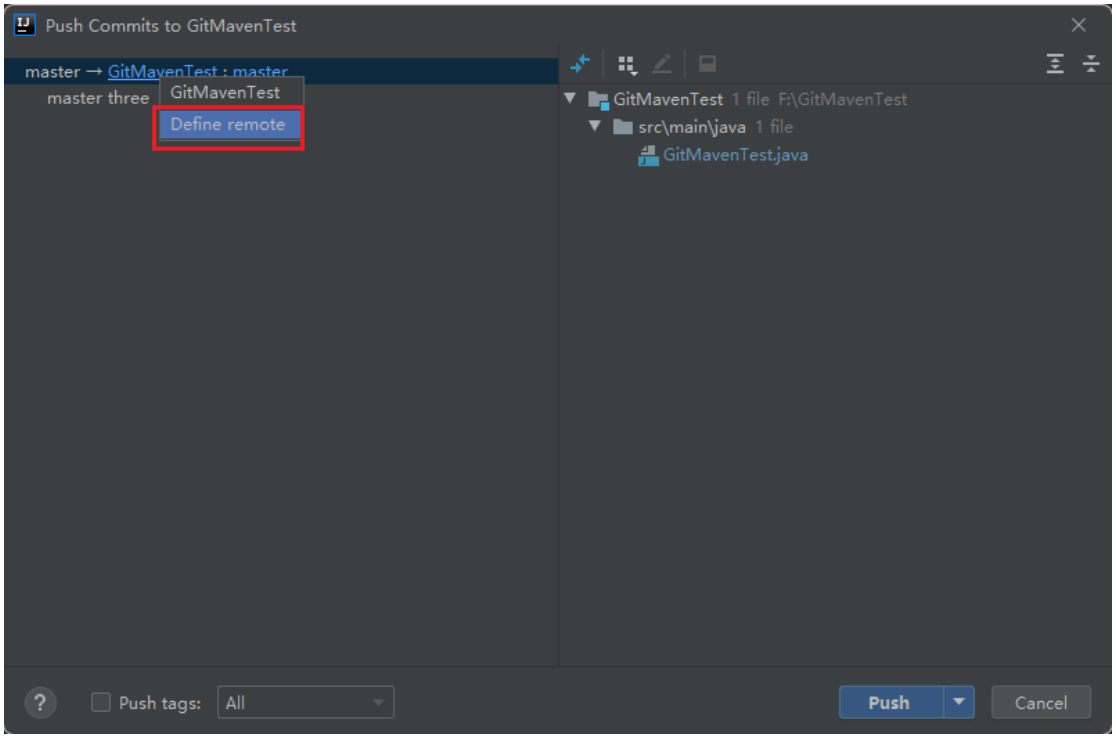

Project
GitMavenTest
.idea
.gitignore
misc.xml
shelf
vcs.xml
GitMavenTest
pom.xml
src
main
java
test
External Libraries
Scratches and Co
pom.xml (GitMavenTest)
GitMavenTest.java
public class GitMavenTest {
public static void main(String[] args) {
System.out.println("git maven test");
System.out.println("git maven test1");
System.out.println("git maven test2");
System.out.println("git maven test3");
System.out.println("master test");
System.out.println("master-1 Test");
System.out.println("push Test");
}
}
New
Add Framework Support...
Cut Ctrl+X
Copy
Paste Ctrl+V
Find Usages Alt+F7
Find in Path... Ctrl+Shift+F
Replace in Path... Ctrl+Shift+R
Analyze
Refactor
Add to Favorites
Reformat Code Ctrl+Alt+L
Optimize Imports Ctrl+Alt+O
Remove Module Delete
Build Module 'GitMavenTest'
Rebuild Module 'GitMavenTest' Ctrl+Shift+F9
Show in Explorer
Directory Path Ctrl+Alt+F12
Open in Terminal
Local History
Git
Reload from Disk
Compare With... Ctrl+D
Open Module Settings F4
Mark Directory as
Remove BOM
Open on GitHub
Create Gist...
Maven
Diagrams
Convert Java File to Kotlin File Ctrl+Alt+Shift+K
Commit Directory...
Add Ctrl+Alt+A
.git/info/exclude
Annotate
Show Current Revision
Compare with the Same Repository Version
Compare with...
Compare with Branch...
Show History
Rollback... Ctrl+Alt+Z
Repository
Git: Log: master-1
Local
master
master-1
Remote
two Class ma Today 19:48
first Class ma Today 19:48
first MyMaven ma Today 19:35
Branches... Ctrl+Shift+`
Tag...
Merge Changes...
Stash Changes...
UnStash Changes...
Reset HEAD...
Remotes...
Clone...
Fetch
Pull...
Push... Ctrl+Shift+K
Rebase...
9: Git
6: TODO
Terminal

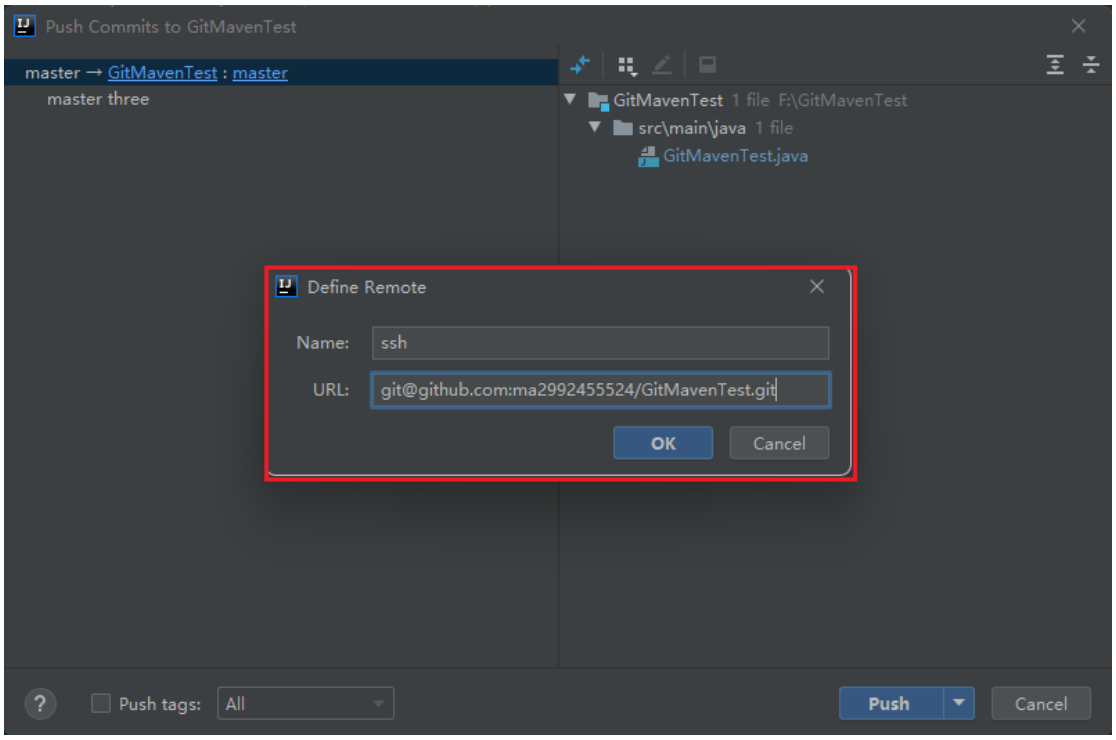
5、默认是 https 登录



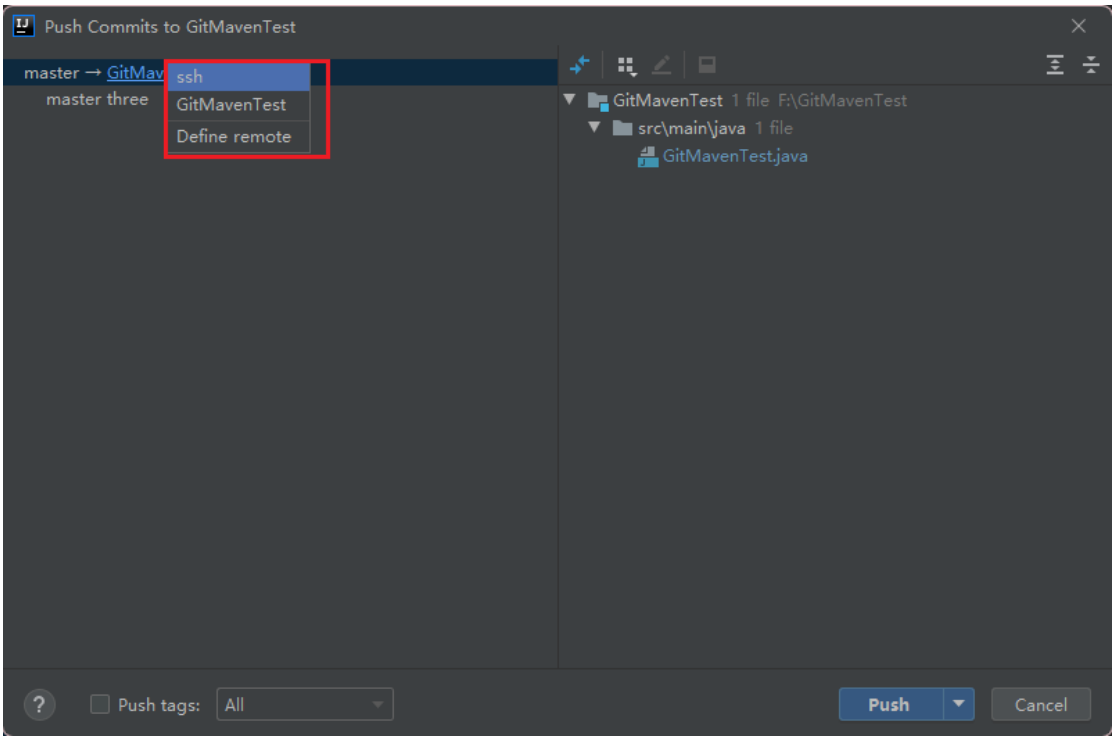
6、设置 ssh 免密登录



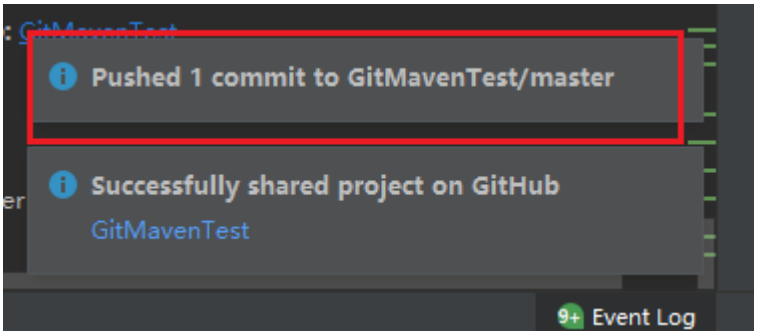

Push Commits to GitMavenTest
master → GitMavenTest : master
master three
GitMavenTest
Define remote
GitMavenTest 1 file F:\GitMavenTest
src\main\java 1 file
GitMavenTest.java
Push tags:
All
Push
Cancel


Push Commits to GitMavenTest
master → GitMavenTest : master
master three
GitMavenTest 1 file F:\GitMavenTest
src\main\java 1 file
GitMavenTest.java
Define Remote
Name:
ssh
URL:
git@github.com:ma2992455524/GitMavenTest.git
OK
Cancel
Push tags:
All
Push
Cancel

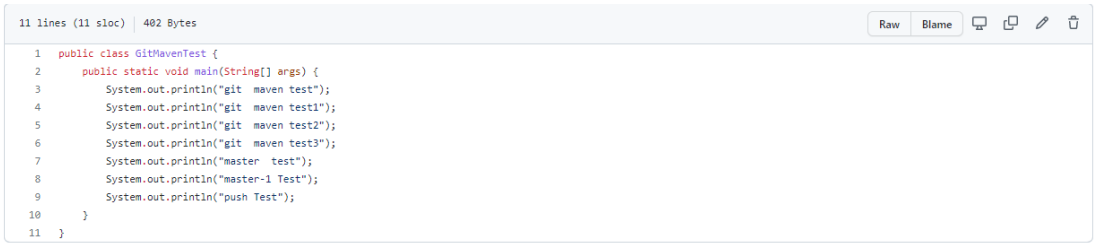
7、添加 ssh 成功



8、push 上传



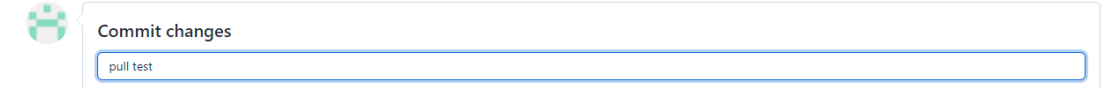
9、查看



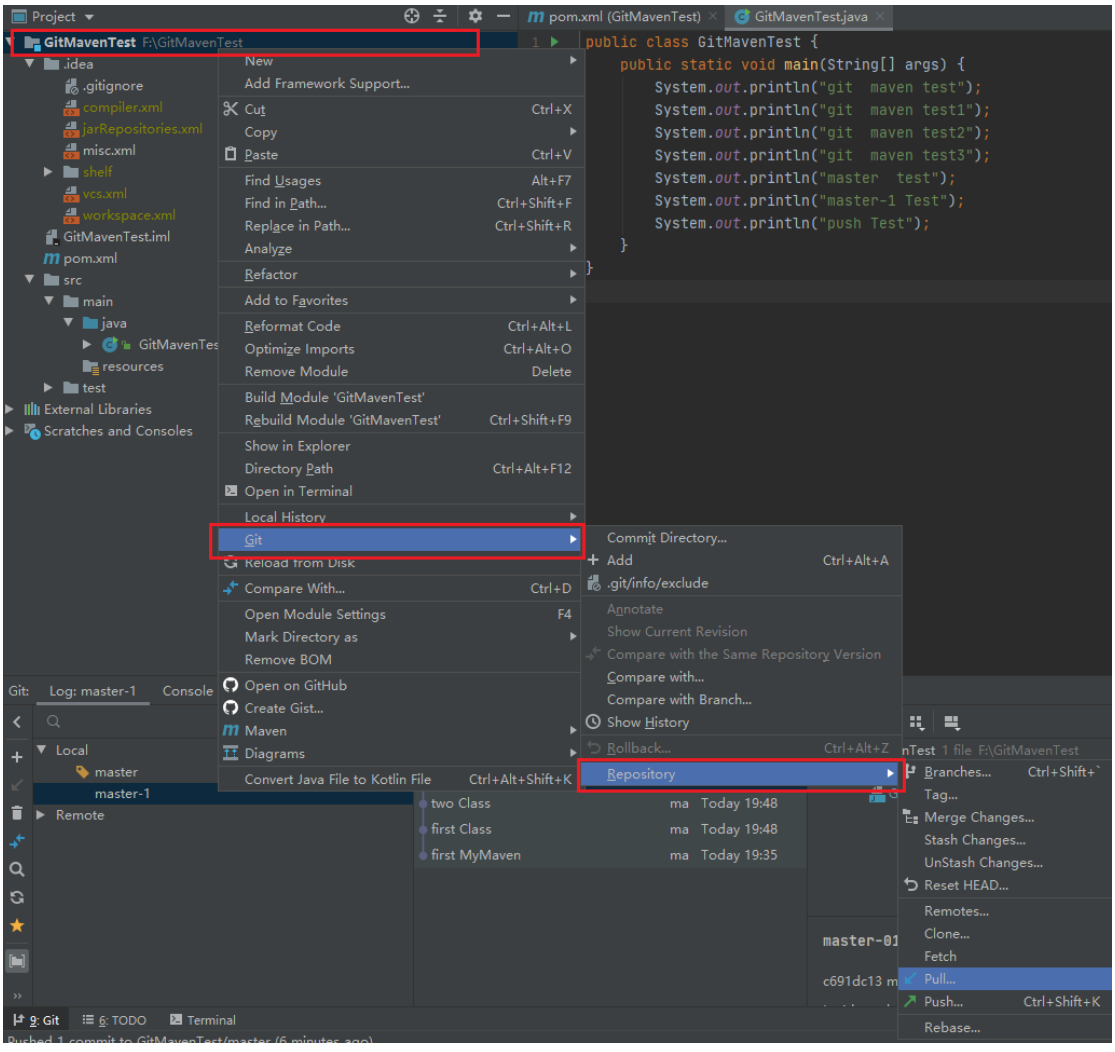
(四)、拉取本地仓库文件

1、修改远程仓库的代码并提交

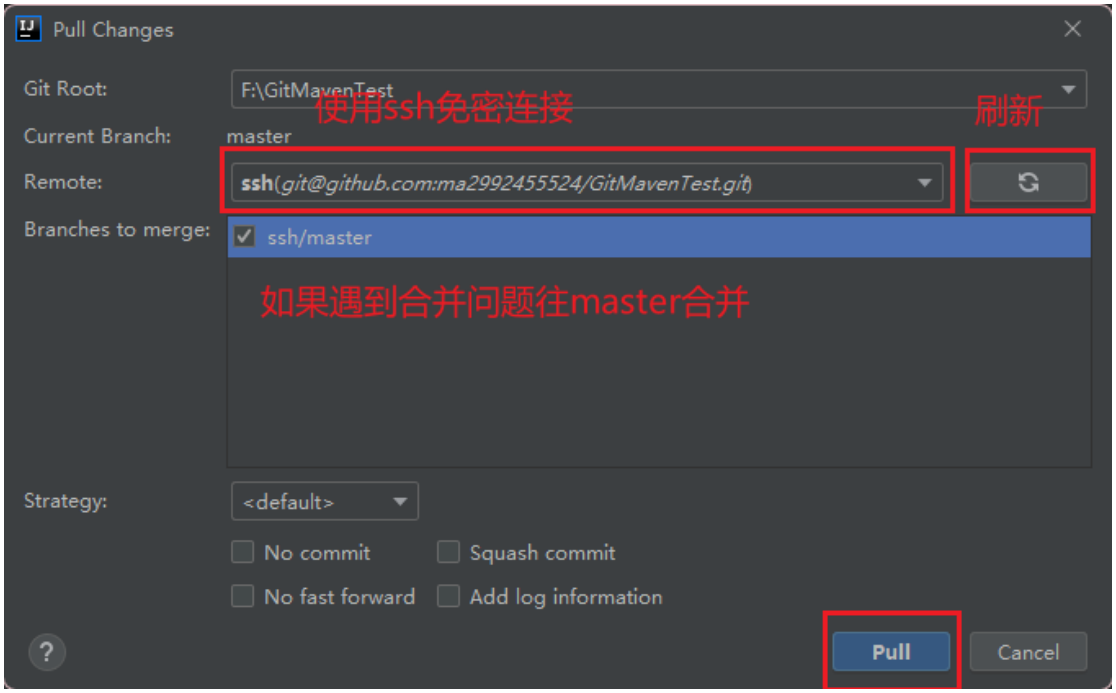
```
public class GitMavenTest {
    public static void main(String[] args) {
        System.out.println("git  maven test");
        System.out.println("git  maven test1");
        System.out.println("git  maven test2");
        System.out.println("git  maven test3");
        System.out.println("master  test");
        System.out.println("master-1 Test");
        System.out.println("push Test");
        System.out.println("pull Test");
    }
}

```



2、使用 IDEA 拉取远程仓库文件



3、拉取窗口

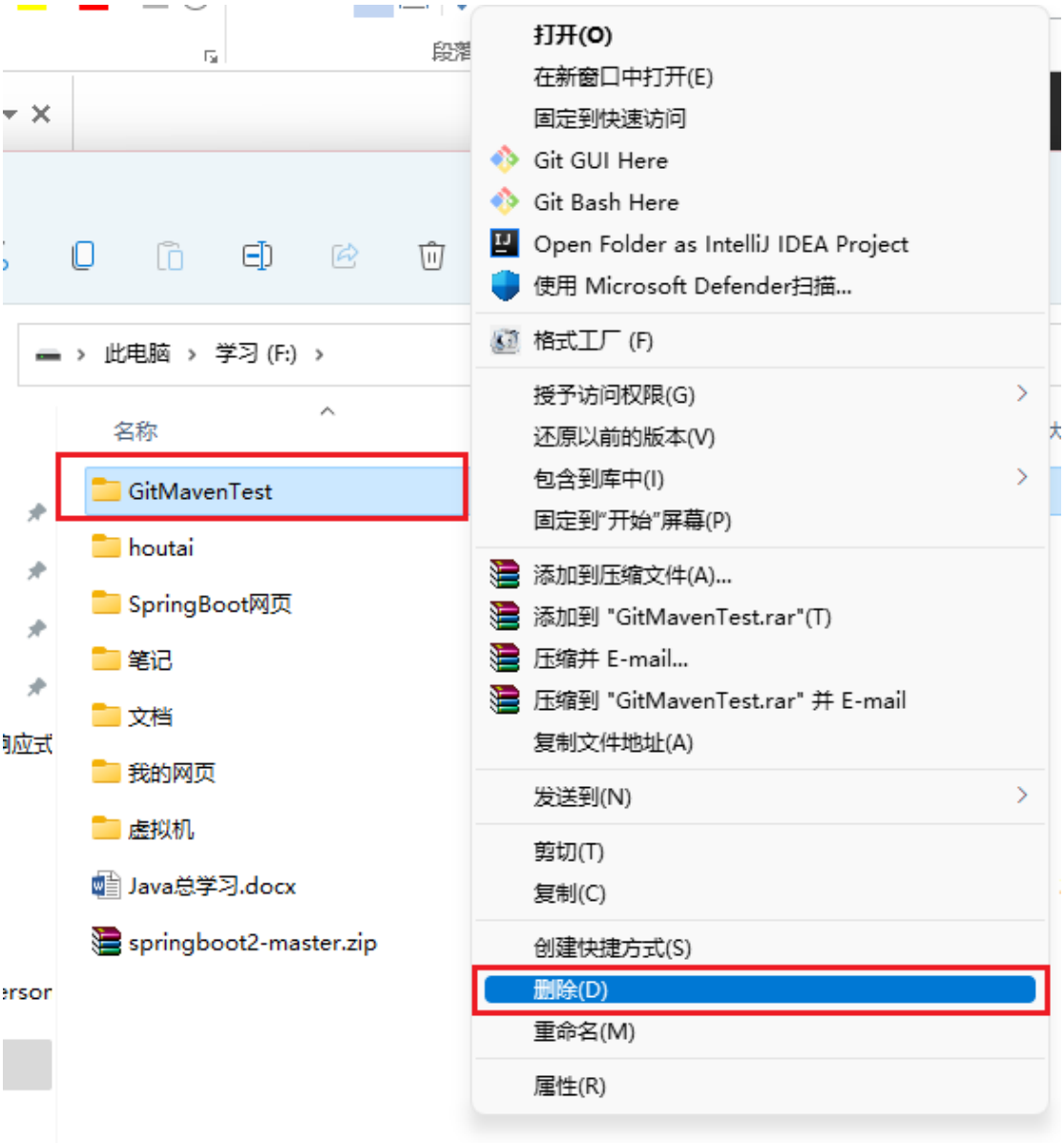


4、拉取成功

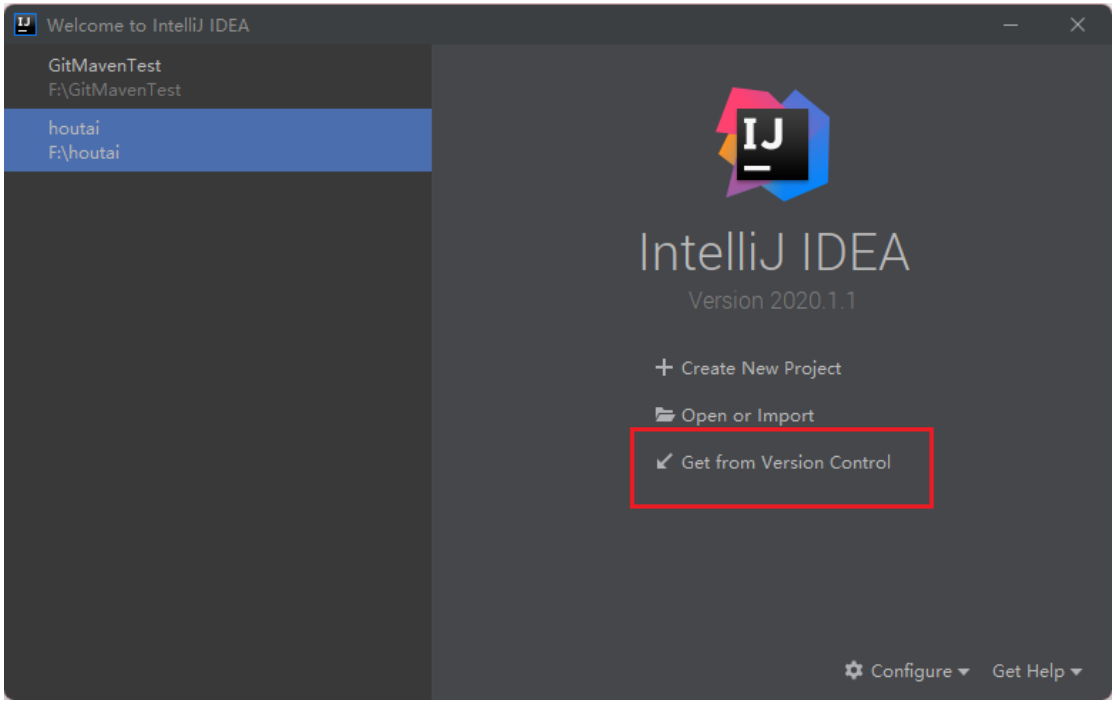
```java
public class GitMavenTest {
    public static void main(String[] args) {
        System.out.println("git  maven test");
        System.out.println("git  maven test1");
        System.out.println("git  maven test2");
        System.out.println("git  maven test3");
        System.out.println("master  test");
        System.out.println("master-1 Test");
        System.out.println("push Test");
        System.out.println("pull Test");
    }
}
```

(五)、克隆远程仓库

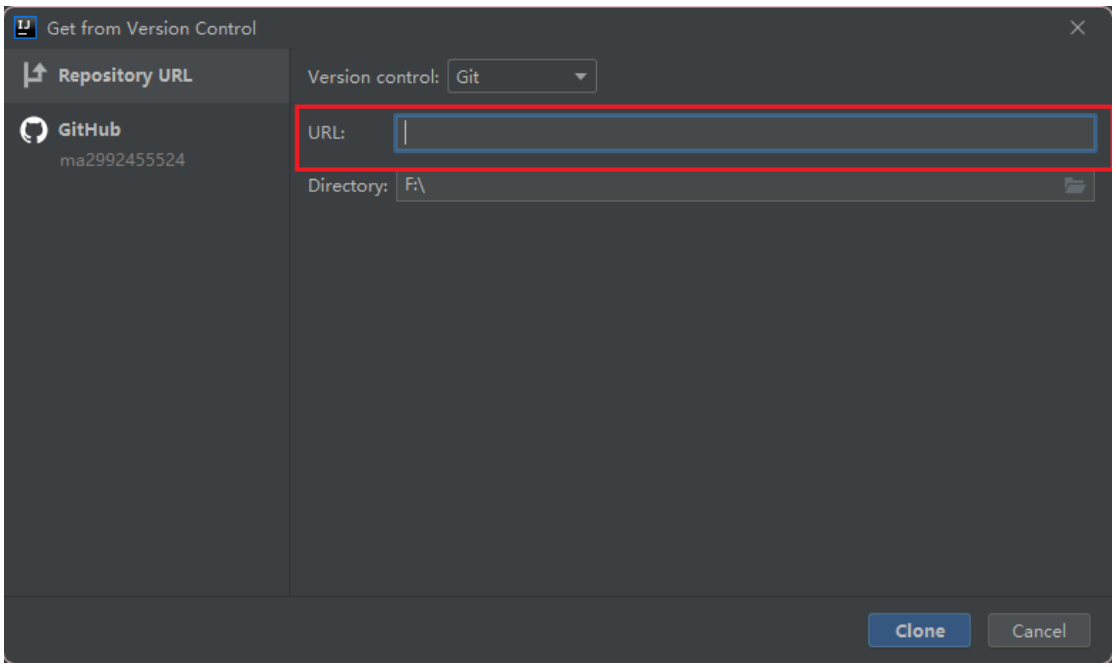
1、首先关闭 IDEA 在本地删除那个项目



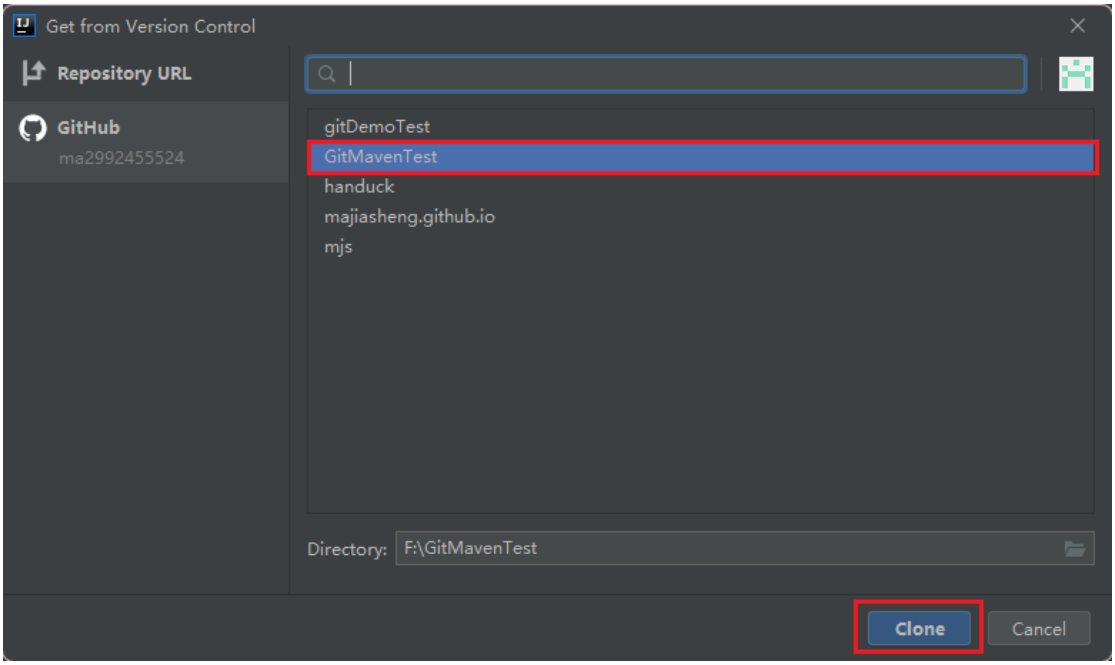
2、打开 IDEA



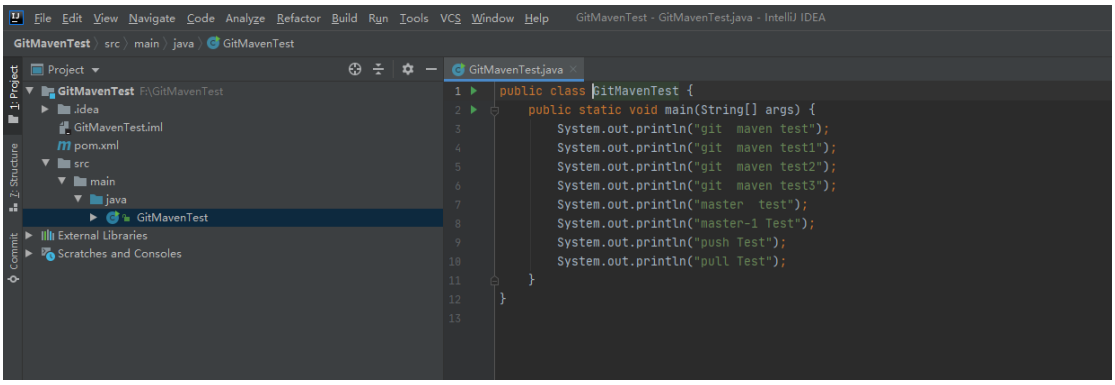
3、使用 URL 获取



4、IDEA 根据登录用户自动扫描远程仓库

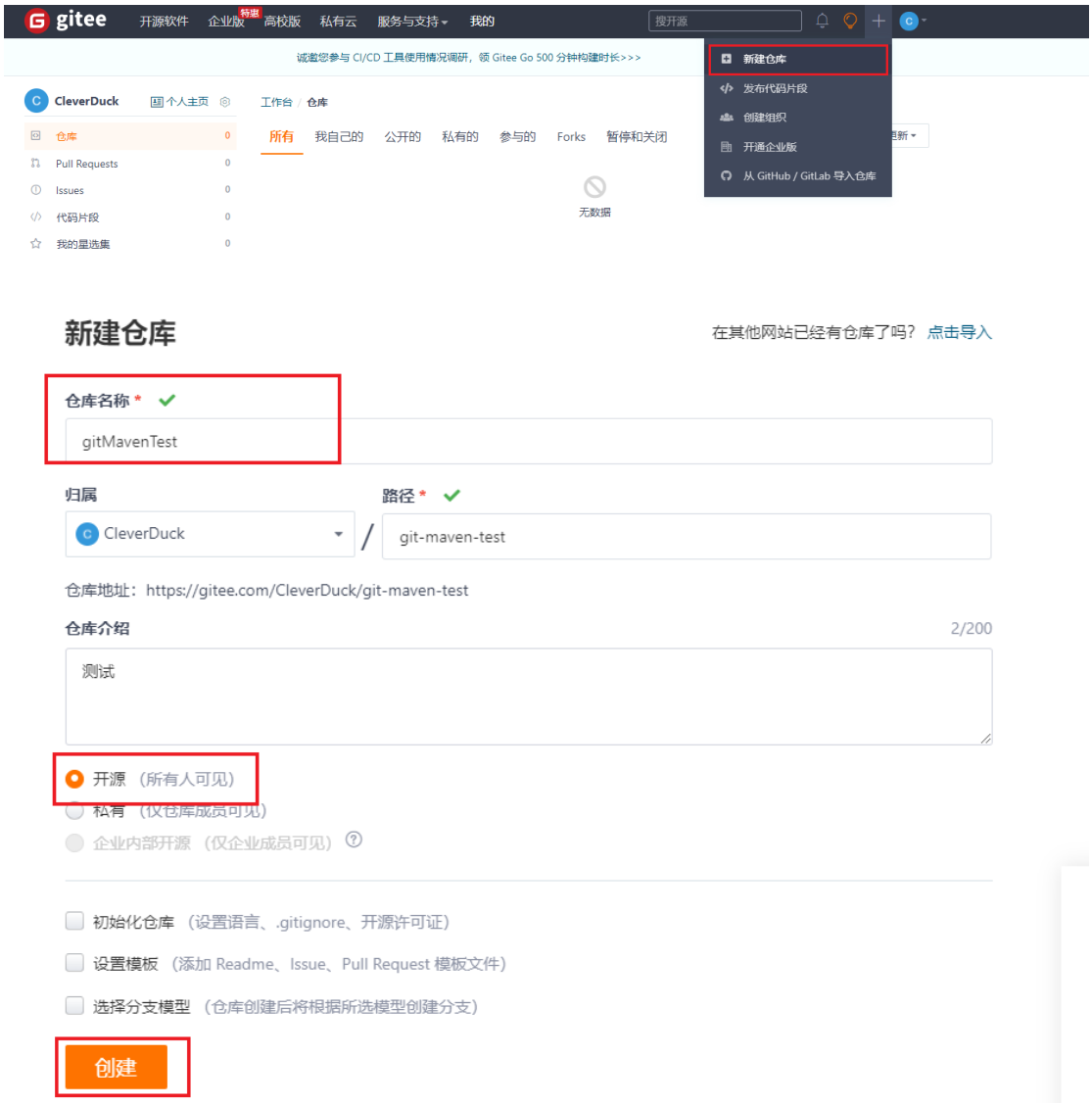


5、克隆成功



六、Gitee

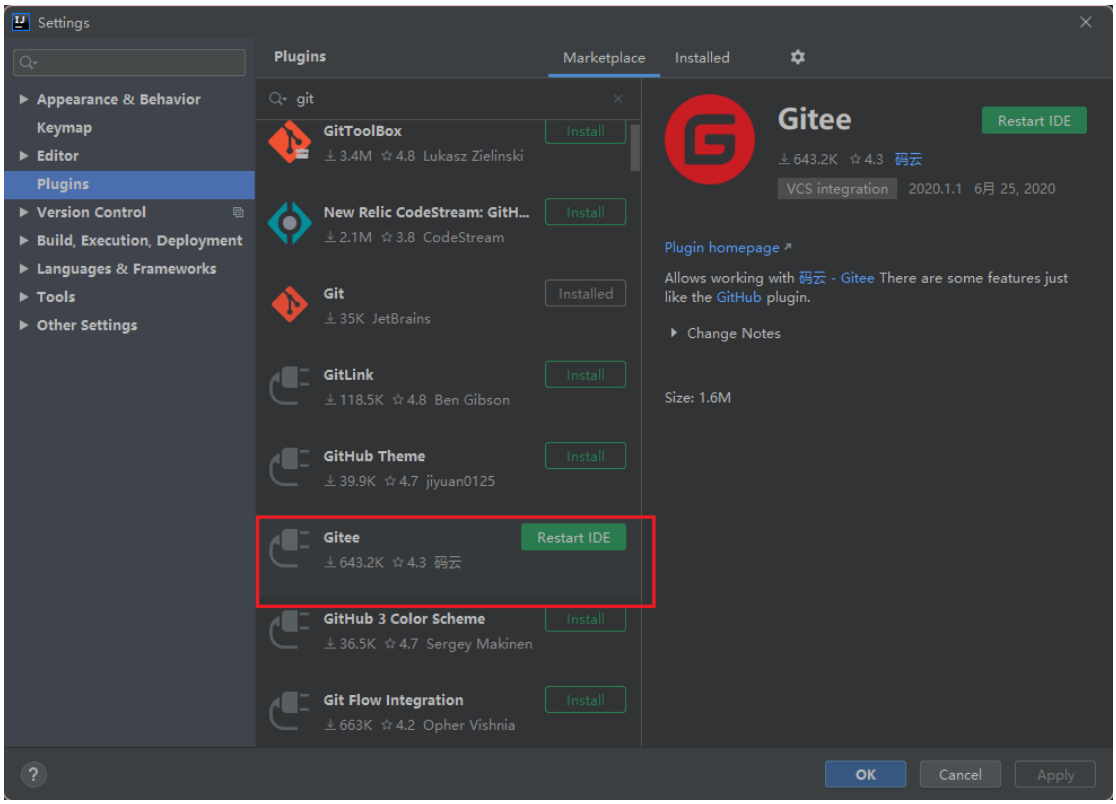
(一)、创建远程仓库



(二)、上传,拉取,克隆

1、码云的上传,拉取,克隆和 github 一模一样

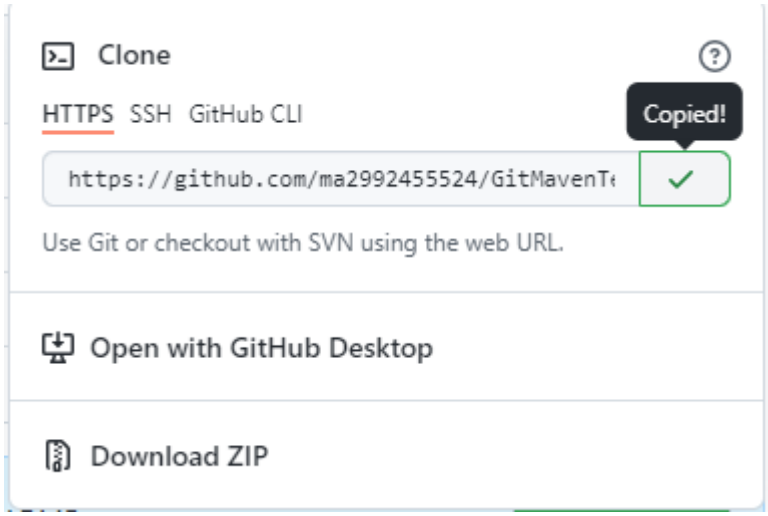
(三)、IDEA 继承 Gitee



(四)、码云导入 github 项目

1、点击创建项目

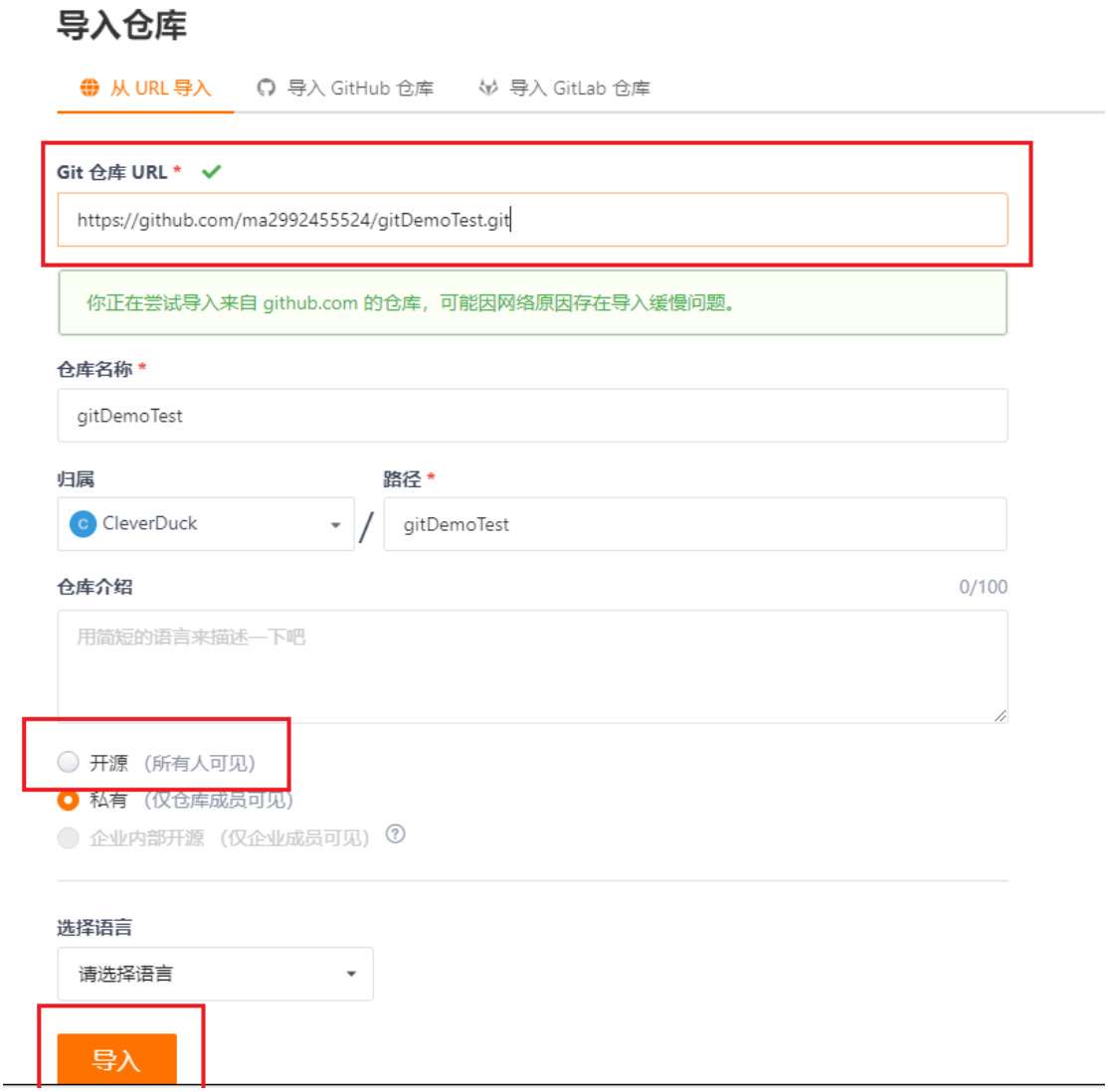
2、复制 github 的 Https 连接



3、添加现有仓库


gitee
开源软件
企业版
特惠
高校版
私有云
服务与支持
我的
搜开源
诚邀您参与 CI/CD 工具使用情况调研，领 Gitee Go 500 分钟构建时长>>>
新建仓库
在其他网站已经有仓库了吗？ 点击导入
仓库名称 *
归属
路径 *
CleverDuck
/
仓库介绍
0/100
用简短的语言来描述一下吧
开源（所有人可见）
私有（仅仓库成员可见）
企业内部开源（仅企业成员可见）
初始化仓库（设置语言、.gitignore、开源许可证）
设置模板（添加 Readme、Issue、Pull Request 模板文件）
选择分支模型（仓库创建后将根据所选模型创建分支）
创建

4、输入连接导入



5、当 github 更新后,更新 gitee 项目

七、GitLab

(一)、安装 gitlab

1、准备一台差不多配置的 Centos

2、将镜像上传到/opt/module 文件中

3、编写脚本

vim gitlab-install.sh
sudo rpm -ivh /opt/module/gitlab-ce-13.10.2-ce.0.el7.x86_64.rpm

sudo yum install -y curl policycoreutils-python openssh-server cronie

sudo lokkit -s http -s ssh

sudo yum install -y postfix

sudo service postfix start

sudo chkconfig postfix on

curl https://packages.gitlab.com/install/repositories/gitlab/gitlabce/script.rpm.sh | sudo bash

sudo EXTERNAL_URL="http://gitlab.example.com" yum -y install gitlabce

4、添加执行权限

chmod –x gitlab-install.sh

5、执行脚本

./gitlab-install.sh

6、初始化 GitLab 服务

gitlab-ctl reconfigure
gitlab Reconfigured!　　出现这个为成功

7、开启&关闭服务

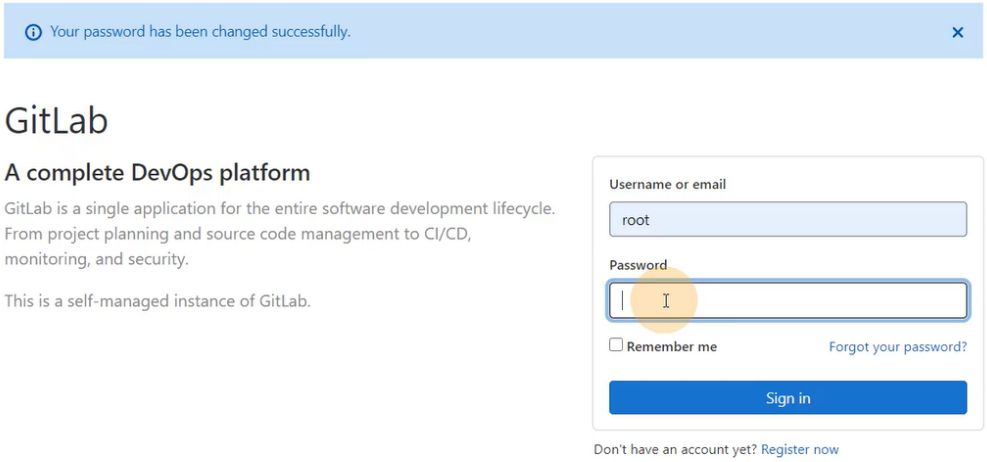
gitlab-ctl start
gitlab-ctl stop

(二)、创建项目

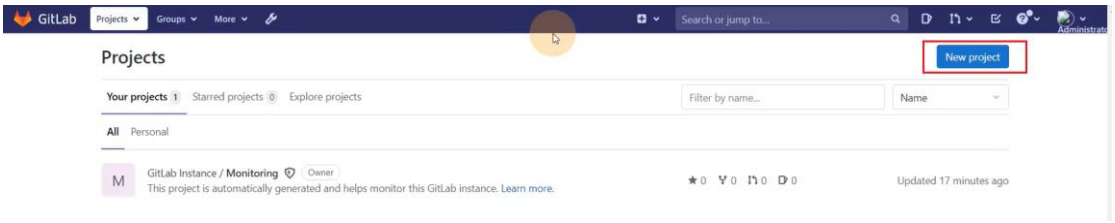
1、登录

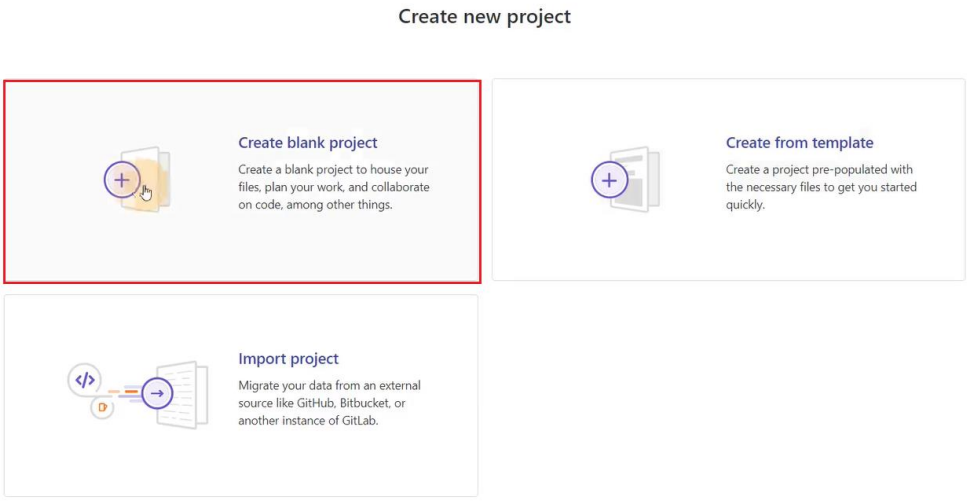
修改本地 host 文件即可域名和 ip 都能访问,默认 80 端口

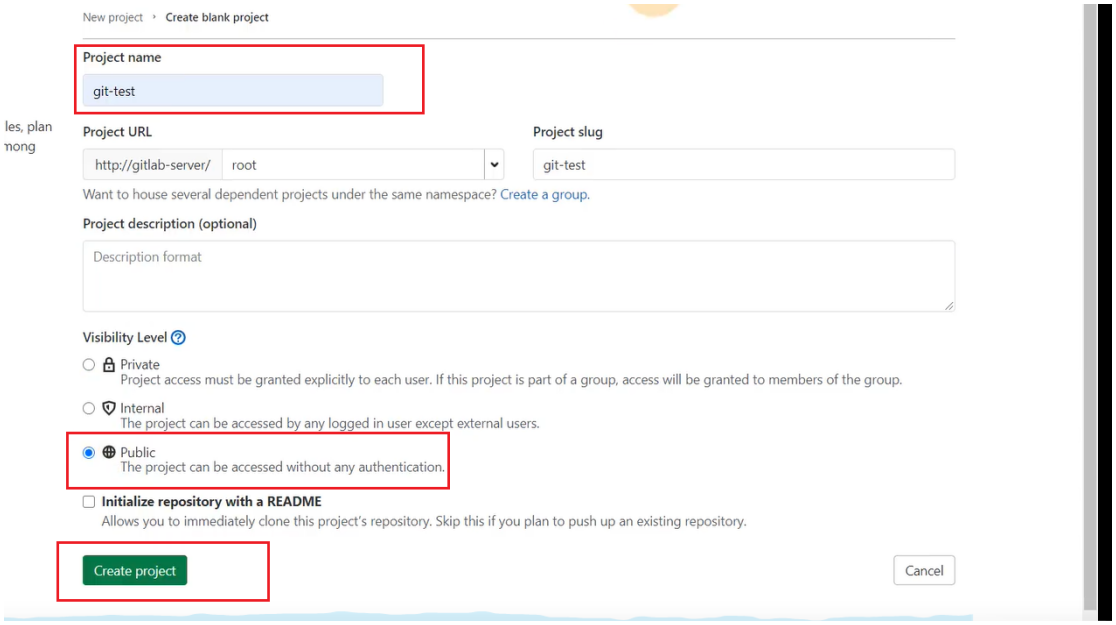
Gitlab 自己会帮你创建一个 root 的账户,当你进行浏览器访问的时候会让你添加密码


Your password has been changed successfully.
GitLab
A complete DevOps platform
GitLab is a single application for the entire software development lifecycle. From project planning and source code management to CI/CD, monitoring, and security.
This is a self-managed instance of GitLab.
Username or email
root
Password
Remember me
Forgot your password?
Sign in
Don't have an account yet? Register now

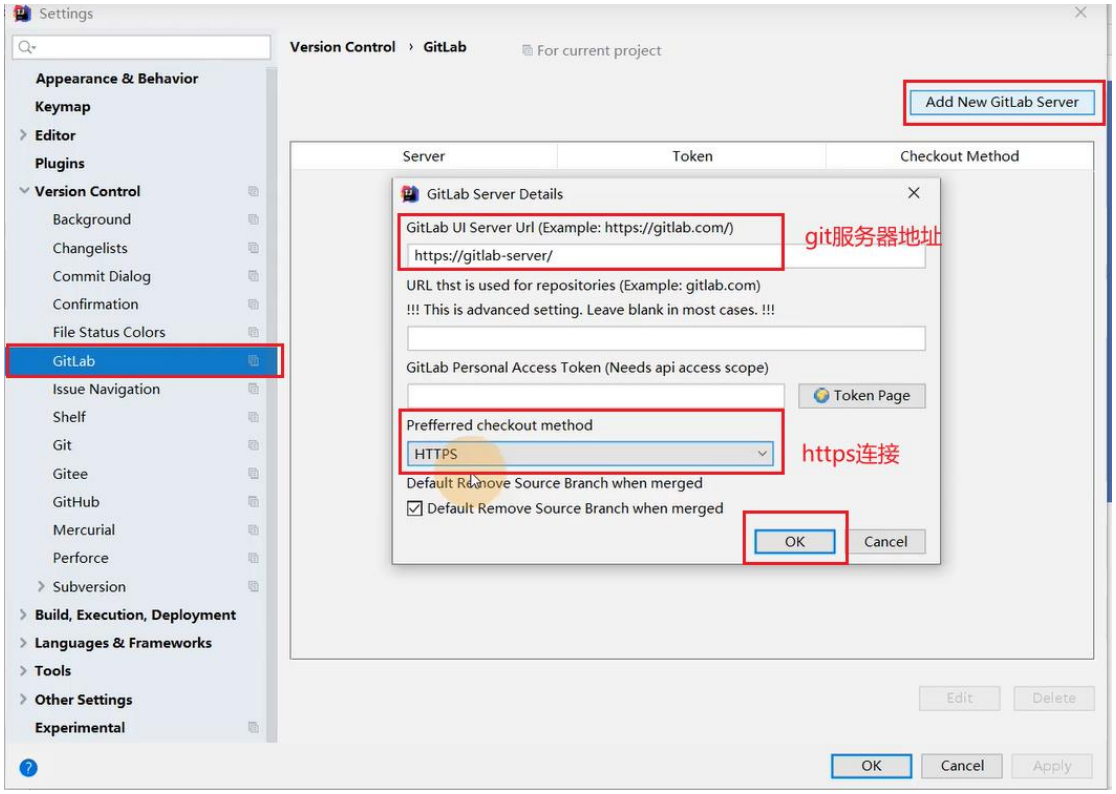
2、创建项目


GitLab
Projects
Groups
More
Search or jump to...
Projects
New project
Your projects 1
Starred projects 0
Explore projects
Filter by name...
Name
All
Personal
M
GitLab Instance / Monitoring
Owner
This project is automatically generated and helps monitor this GitLab instance. Learn more.
Updated 17 minutes ago

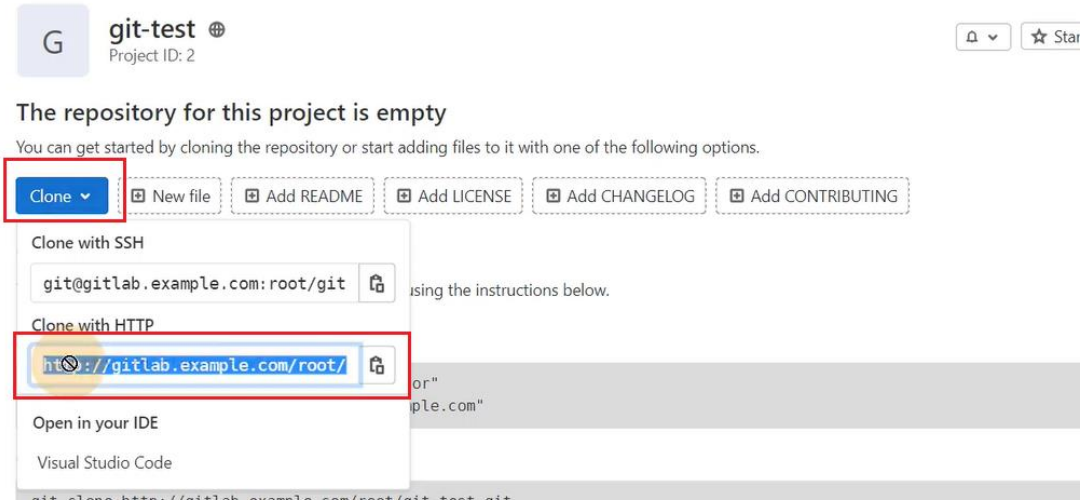

Create new project
Create blank project
Create a blank project to house your files, plan your work, and collaborate on code, among other things.
Create from template
Create a project pre-populated with the necessary files to get you started quickly.
Import project
Migrate your data from an external source like GitHub, Bitbucket, or another instance of GitLab.



(三)、IDEA 集成 GitLab



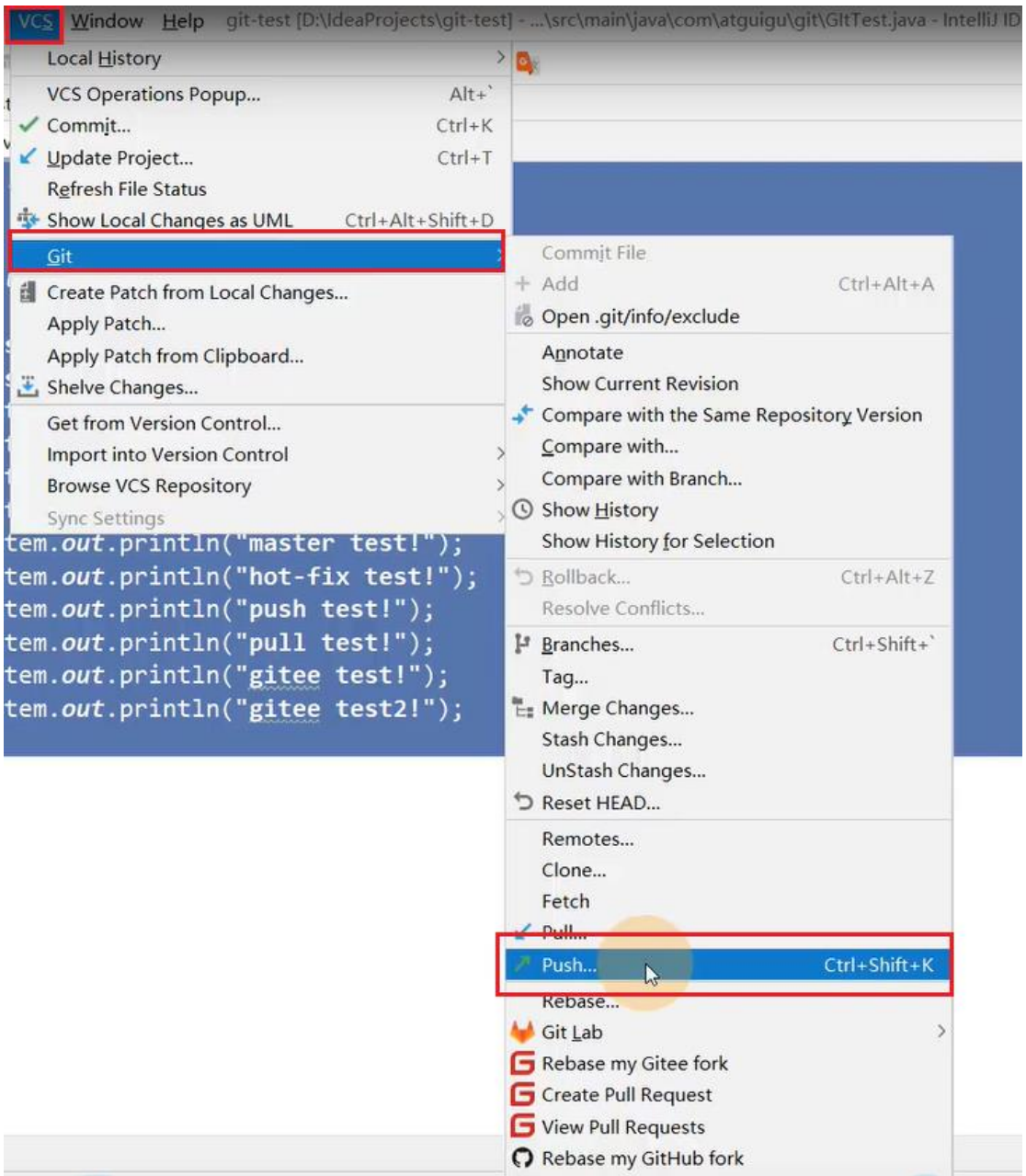
3、复制链接



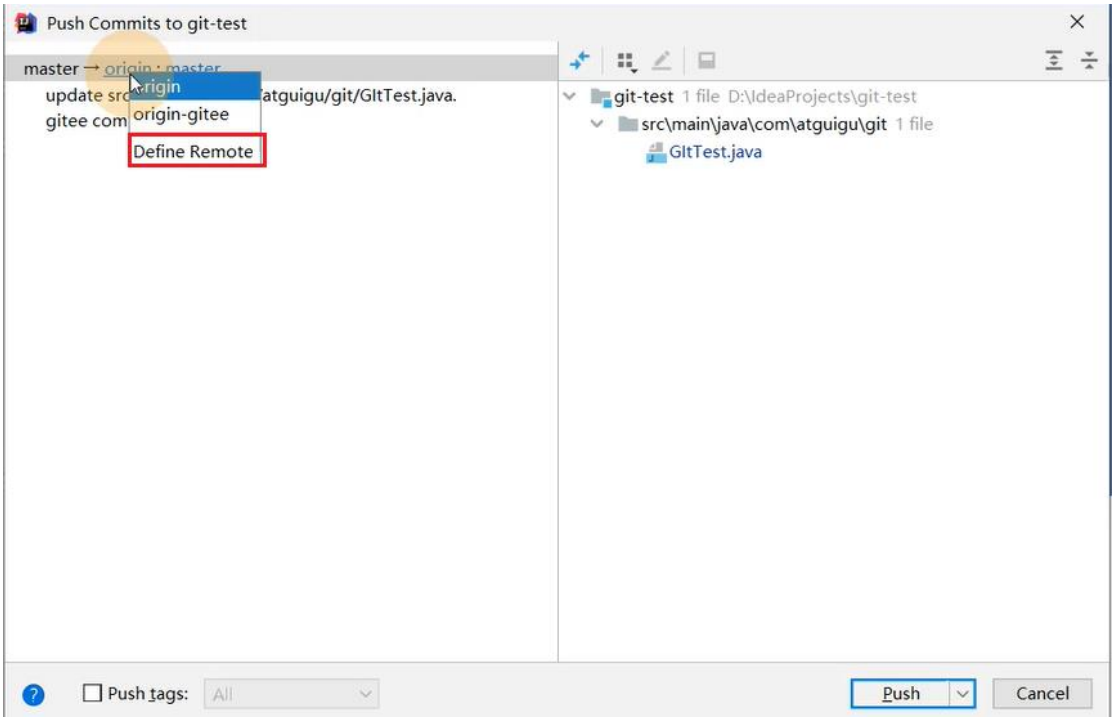
4、修改链接(默认给的是一个模板,修改成自己的链接)

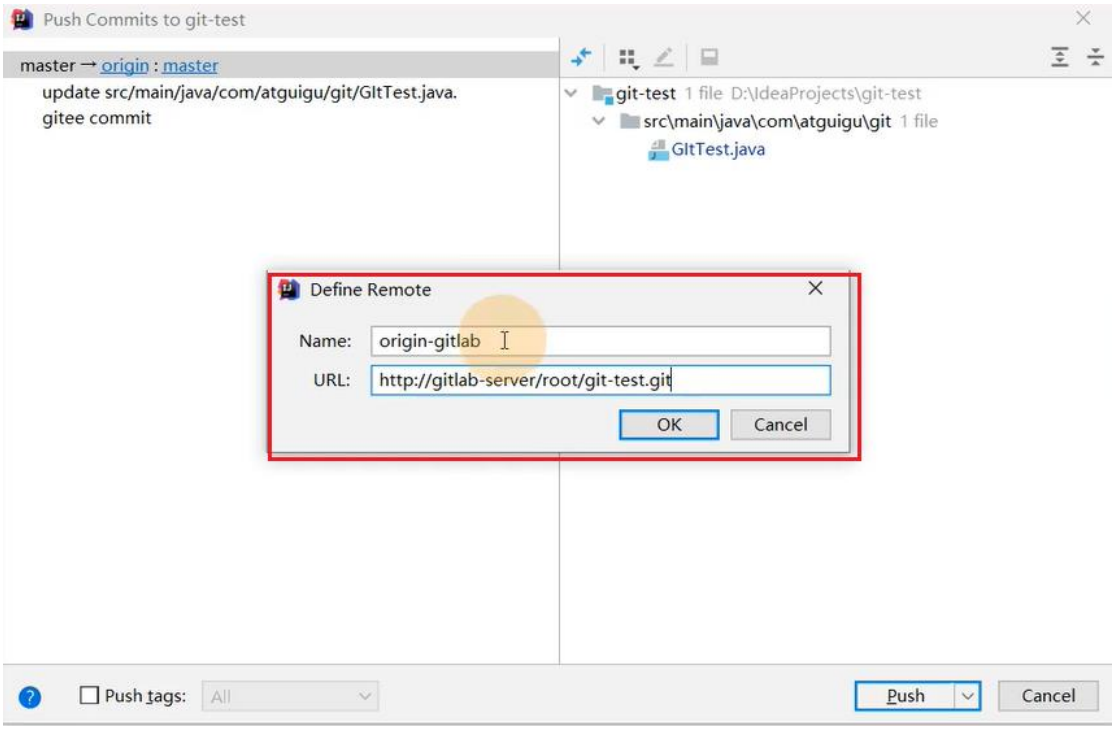
http://gitlab-server/root/git-test.git

5、IDEA 连接 GitLab



6、自定义连接


Push Commits to git-test
master → origin : master
update src/main/java/com/atguigu/git/GItTest.java.
gitee com
origin
origin-gitee
Define Remote
git-test 1 file D:\IdeaProjects\git-test
src\main\java\com\atguigu\git 1 file
GItTest.java
Push tags:
All
Push
Cancel


Push Commits to git-test
master → origin : master
update src/main/java/com/atguigu/git/GItTest.java.
gitee commit
git-test 1 file D:\IdeaProjects\git-test
src\main\java\com\atguigu\git 1 file
GItTest.java
Define Remote
Name:
origin-gitlab
URL:
http://gitlab-server/root/git-test.git
OK
Cancel
Push tags:
All
Push
Cancel

(三)、上传,拉取,克隆

1、都和 github 一模一样