

在 VSCode 中使用 Git 进行项目版本控制

一. 版本控制

- 本节的主要目的是为了让读者了解版本控制的一些主要内容，包括版本控制的概念，意义以及常见的版本控制工具。通过本节内容，读者对版本控制将会有初步的了解。

1. 什么叫版本控制

- 版本控制是指对软件开发过程中各种程序代码、配置文件及说明文档等文件变更的管理，是软件配置管理的核心思想之一。

2. 为什么要进行版本控制

- 在软件开发过程，每天都会产生新的代码，代码合并的过程中可能会出现如下问题：
 - 代码被覆盖或丢失
 - 代码写的不理想希望还原之前的版本
 - 希望知道与之前版本的差别
 - 是谁修改了代码以及为什么修改
 - 发版时希望分成不同的版本(测试版、发行版)
- 在软件开发的过程中，只需使用版本控制工具，就能解决上述的问题，这就是为什么要进行版本控制的原因。

3. 版本控制的主要功能

- 可以随时回滚到之前的版本
- 协同开发时不会覆盖别人的代码
- 留下修改记录，以便随时查看
- 发版时可以方便的管理不同的版本

4. 常见的版本控制工具

目前常用的版本控制工具主要是 SVN 和 Git：

(1) SVN

- SVN 是 subversion 的缩写，是一个开放源代码的版本控制系统，通过采用分支管理系统的高效管理，简而言之就是用于多个人共同开发同一个项目，实现共享资源，实现最终集中式的管理。

(2) Git

- Git 是 Linus Trovalds 大神的作品，是一个开放源码的版本控制软件。与 SVN 最大的区别，就是分布式的管理。

(3) 使用选择

- 如果对访问控制、权限分配和代码安全性等要求比较高的，建议使用 SVN。
- 如果是分布式，多人开发，版本迭代比较快的项目，建议使用 Git。

二. 工具介绍和使用

- 本节的主要目的是为了让读者了解在版本控制中常用的两个工具 Git 和 GitHub。通过本节内容，读者将会了解 Git 的基本使用以及了解 GitHub 的作用。

1. Git

(1) Git 简介

Git (读音为 /git/) 是一个开源的分布式版本控制系统，可以有效、高速地处理从很小到非常大的项目版本管理。

(2) Git 安装

- [下载 git](#)，打开即可自动下载

Downloading Git



Your download is starting...

You are downloading the latest (**2.24.1**) **64-bit** version of **Git for Windows**. This is the most recent **maintained build**. It was released **15 days ago**, on 2019-12-10.

If your download hasn't started, [click here to download manually](#).

Other Git for Windows downloads

Git for Windows Setup

32-bit Git for Windows Setup.

64-bit Git for Windows Setup.

根据电脑配置选择相应的版本

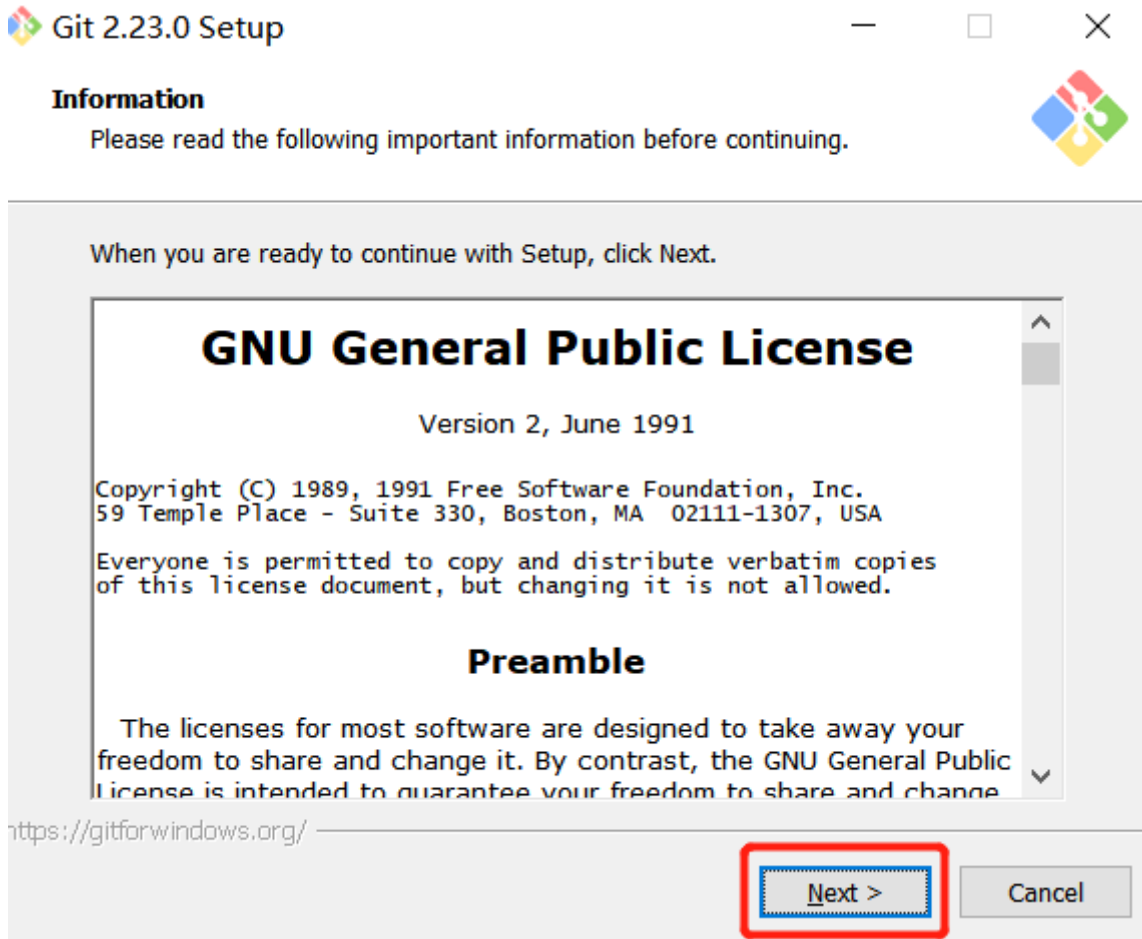
Git for Windows Portable ("thumbdrive edition")

32-bit Git for Windows Portable.

64-bit Git for Windows Portable.

The current source code release is version **2.24.1**. If you want the newer version, you can build it from [the source code](#).

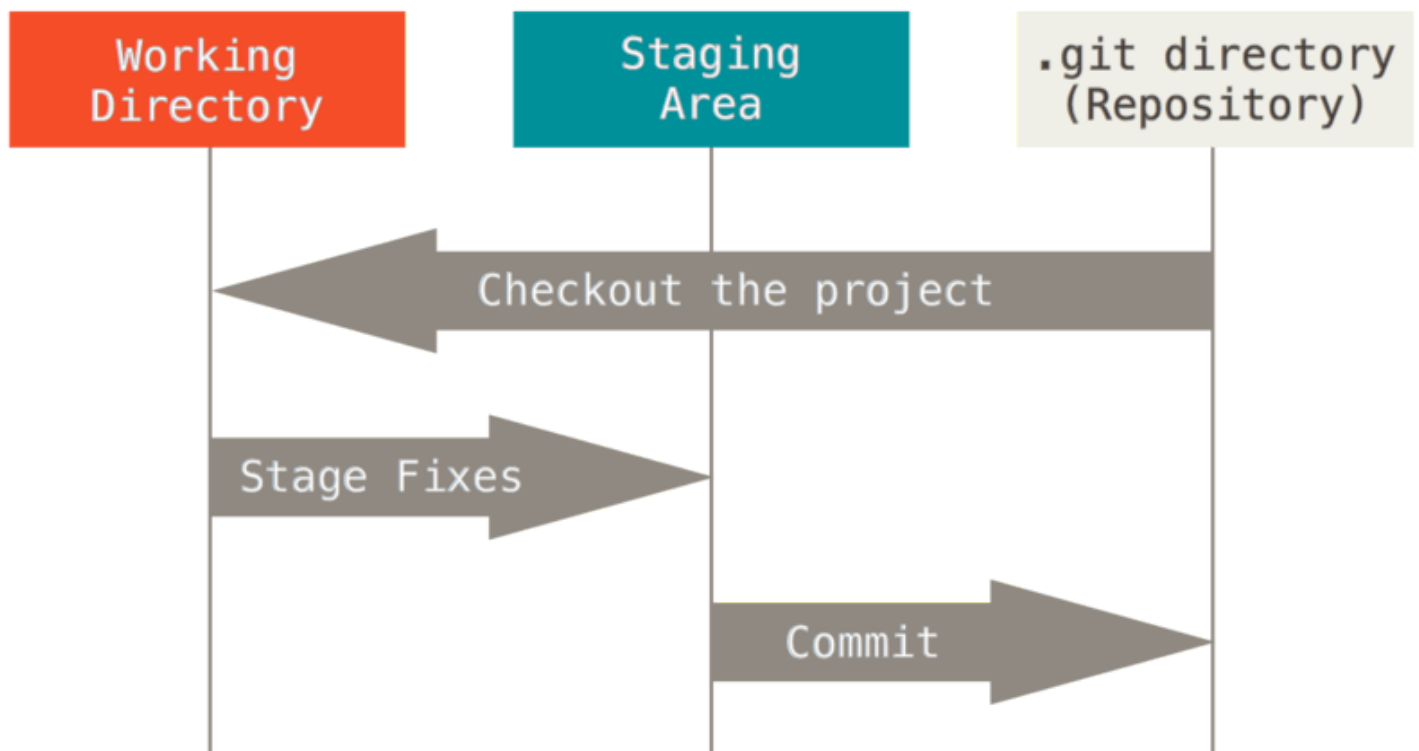
- 如果没有特殊要求，一路点击 next 即可



- [Git 安装选项说明](#)

(3) Git 三种状态

Git 有三种状态，你的文件可能处于其中之一：已提交（committed）、已修改（modified）和已暂存（staged）。已提交表示数据已经安全的保存在本地数据库中。已修改表示修改了文件，但还没保存到数据库中。已暂存表示对一个已修改文件的当前版本做了标记，使之包含在下次提交的快照中。




- Git 仓库目录是 Git 用来保存项目的元数据和对象数据库的地方。这是 Git 中最重要的部分，从其它计算机克隆仓库时，拷贝的就是这里的数据。
- 工作目录是对项目的某个版本独立提取出来的内容。这些从 Git 仓库的压缩数据库中提取出来的文件，放在磁盘上供你使用或修改。
- 暂存区域是一个文件，保存了下次将提交的文件列表信息，一般在 Git 仓库目录中。有时候也被称作“索引”，不过一般说法还是叫暂存区域。

(4) Git 常见命令

- git init
用 git init 在目录中创建新的 Git 仓库。你可以在任何时候、任何目录中这么做，完全是本地化的。
在目录中执行 git init，就可以创建一个 Git 仓库了。比如我们创建 example 项目：
 - 新建文件夹:example
 - 在文件夹 example 下打开 Git Bash，输入 git init

名称	修改日期	类型
 .git	2019/12/26 13:34	文件夹

 MINGW64:/e/example

```
LWL@DESKTOP-60RRPGS MINGW64 /e/example
$ git init
Initialized empty Git repository in E:/example/.git/


LWL@DESKTOP-60RRPGS MINGW64 /e/example (master)
$
```

你可以看到在你的项目中生成了 .git 这个子目录。这就是你的 Git 仓库了，所有有关你的此项目的快照数据都存放在这里。

- git add

git add 命令可将该文件添加到暂存区域，如我们在 example 项目里面添加 README.txt 文件。此时，该文件只是存储在工作目录。我们需要用 git add README.txt 命令，将文件添加到 Git 暂存区域。

名称	修改日期	类型
 .git	2019/12/26 13:51	文件夹
 README.txt	2019/12/26 13:50	文本文档

 MINGW64:/e/example

```
LWL@DESKTOP-60RRPGS MINGW64 /e/example (master)
$ git add README.txt

LWL@DESKTOP-60RRPGS MINGW64 /e/example (master)
$ git status -s
A README.txt

LWL@DESKTOP-60RRPGS MINGW64 /e/example (master)
$
```

新项目中，添加所有文件很普遍，我们可以使用 git add . 命令来添加当前项目的所有文件。

- git status

git status 命令用于查看项目的当前状态。

该命令的时候加了 -s 参数，以获得简短的结果输出


- git commit

使用 git add 命令将想要快照的内容写入暂存区域，而执行 git commit 将暂存区域的内容添加到

Git 仓库中。

接下来，我们使用 `git commit` 命令将暂存区域的内容添加到 Git 仓库中。

名称	修改日期	类型
 .git	2019/12/26 13:58	文件夹
 README.txt	2019/12/26 13:50	文本文档

 MINGW64:/e/example

```
LWL@DESKTOP-60RRPGS MINGW64 /e/example (master)
$ git add README.txt

LWL@DESKTOP-60RRPGS MINGW64 /e/example (master)
$ git status -s
A README.txt

LWL@DESKTOP-60RRPGS MINGW64 /e/example (master)
$ git commit -m "Add Files"
[master (root-commit) 8110fda] Add Files
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 README.txt

LWL@DESKTOP-60RRPGS MINGW64 /e/example (master)
$
```

使用 `-m` 选项以在命令行中提供提交注释。

如果你觉得 `git add` 提交缓存的流程太过繁琐，Git 也允许你用 `-a` 选项跳过这一步。

此时我修改了项目中的两个文件，使用 `git commit -am` 命令

名称	修改日期	类型
.git	2019/12/26 14:02	文件夹
example.txt	2019/12/26 14:00	文本文档
README.txt	2019/12/26 14:01	文本文档

MINGW64:/e/example

```
LWL@DESKTOP-60RRPGS MINGW64 /e/example (master)
$ git status -s
M README.txt
?? example.txt

LWL@DESKTOP-60RRPGS MINGW64 /e/example (master)
$ git commit -am "Add Files"
[master a61c58f] Add Files
1 file changed, 1 insertion(+)

LWL@DESKTOP-60RRPGS MINGW64 /e/example (master)
$ |
```

从图中可以看出，git commit -am 命令等于 git add . 命令加上 git commit -m 命令。

2. GitHub

(1) GitHub 简介

GitHub是一个面向开源及私有软件项目的托管平台，因为只支持 Git 作为唯一的版本库格式进行托管，故名 GitHub。

(2) GitHub 作用

GitHub可以托管各种 Git 库，并提供一个 web 界面。

三. 计算器项目范例

- 本节将以计算器项目为范例（此计算器项目目前只是一个简单的软件项目，暂未加入版本控制功能，暂未存储到 GitHub 仓库），讲解 VSCode，Git，GitHub 联合使用前的一些准备以及从零演示如何将此项目加入版本控制功能和存储到远程 GitHub 仓库。本节主要分两部分进行讲解：
 1. VSCode，Git，GitHub 联合使用前的准备工作
 2. 将项目加入版本控制功能和存储到远程 GitHub 仓库的基本步骤

1. 必要准备

在演示之前，我们需要做一些准备工作，不然无法将 VSCode，GitHub 和 Git 联合使用。这些准备工作，只需进行一次即可，以后进行项目版本控制无需再做。

准备工作主要有 3 部分：

1. Git 初始化配置
2. 添加 SSH Keys
3. 在 VSCode 中使用 Git工具配置

(1) Git 初始化配置

在 Git Bash 命令行中输入：

```
git config --global user.name "Your Name"
git config --global user.email "email@example.com"
```

因为Git是分布式版本控制系统，所以，每个机器都必须自报家门：你的名字和Email地址。

(2) 添加 SSH Keys

SSH Keys 能让你方便的登录到 SSH 服务器，而无需输入密码。

- 1). 打开 Git Bash
- 2). 在 Git Bash 中输入 `ssh-keygen -t rsa -C "your email address"`，按三次回车，会生成一个 `id_rsa.pub`文件（生成位置请看 Git Bash 的输出信息），里面记录着 SSH 秘钥。
- 3). 用记事本打开 `id_rsa.pub` 文件，复制里面信息。
- 4). 进入网址 <https://github.com/settings/ssh/new>（先要登录网站），然后填写信息。其中Title随便输入内容，Key 为我们刚才复制的内容

SSH keys / Add new

Title

Key

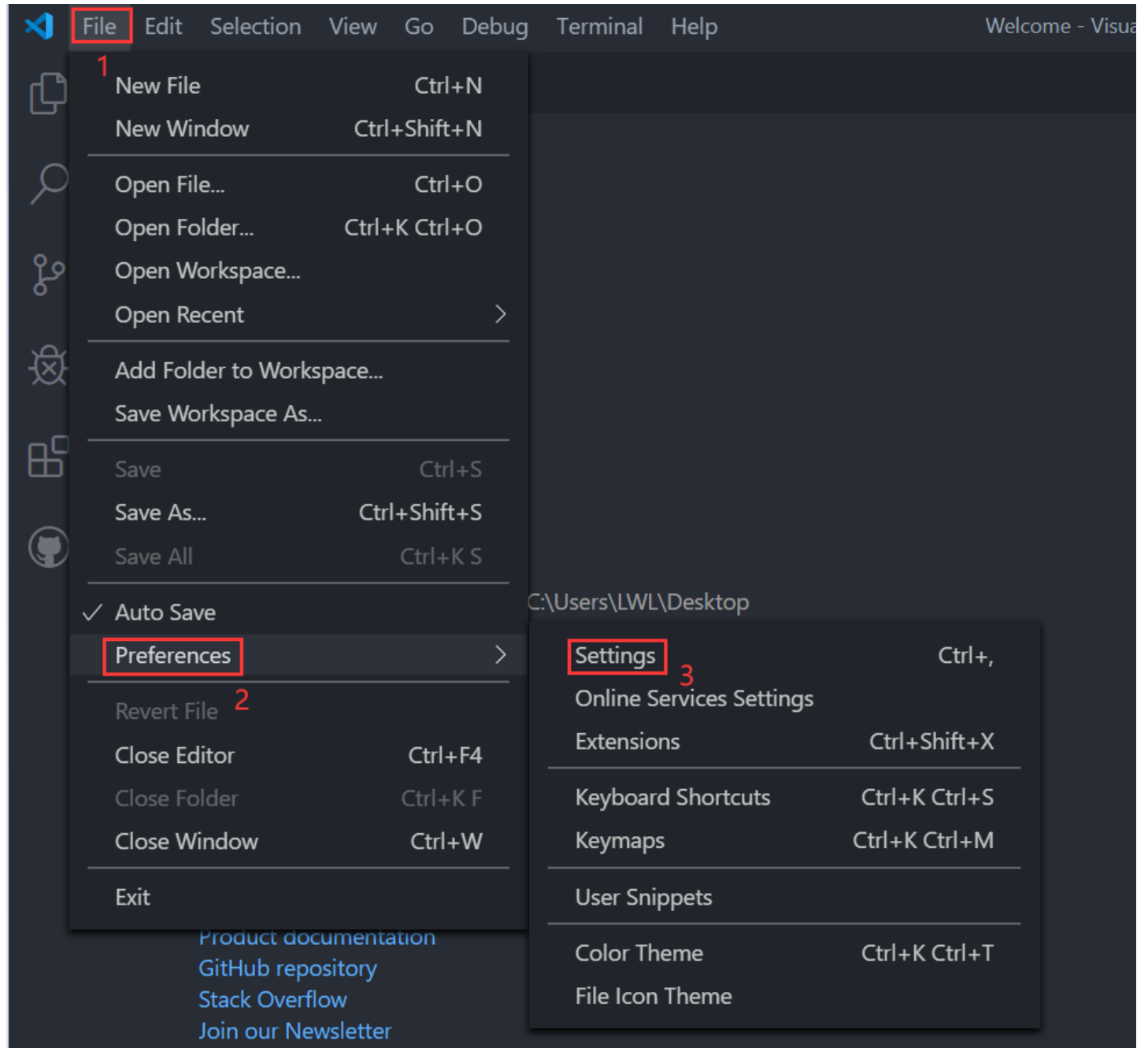
Begins with 'ssh-rsa', 'ssh-ed25519', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', or 'ecdsa-sha2-nistp521'

Add SSH key

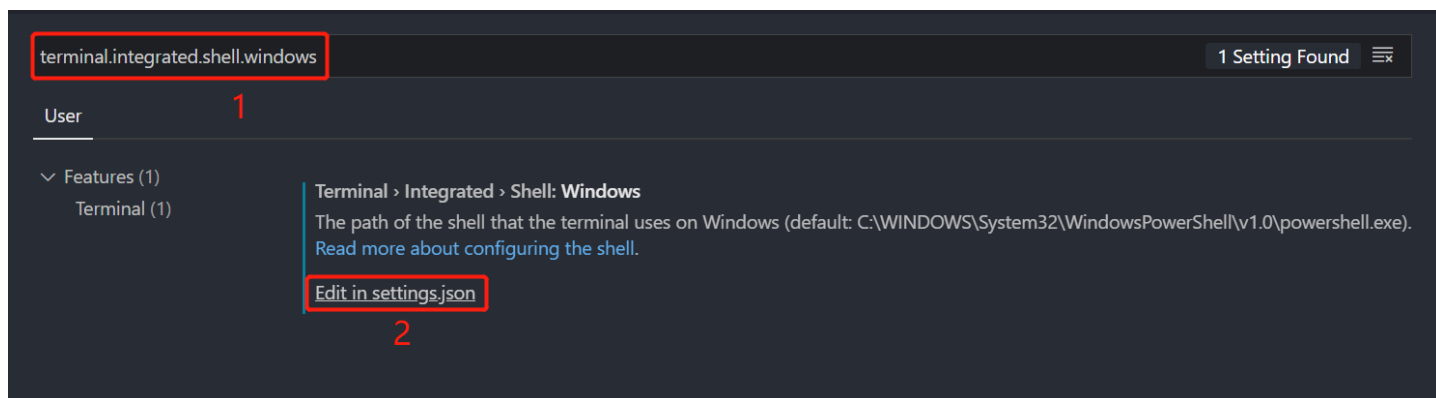
5). 信息填写完成之后点击 Add SSH key。

(3) 在 VSCode 中使用 Git工具配置

1). 打开 VSCode ， 点击 File， 再点击 Preferences ， 最后点击 Settings



2). 在搜索框输入：terminal.integrated.shell.windows ， 输入完成之后， 点击 Edit in settings.json ， 进入 settings json 页面

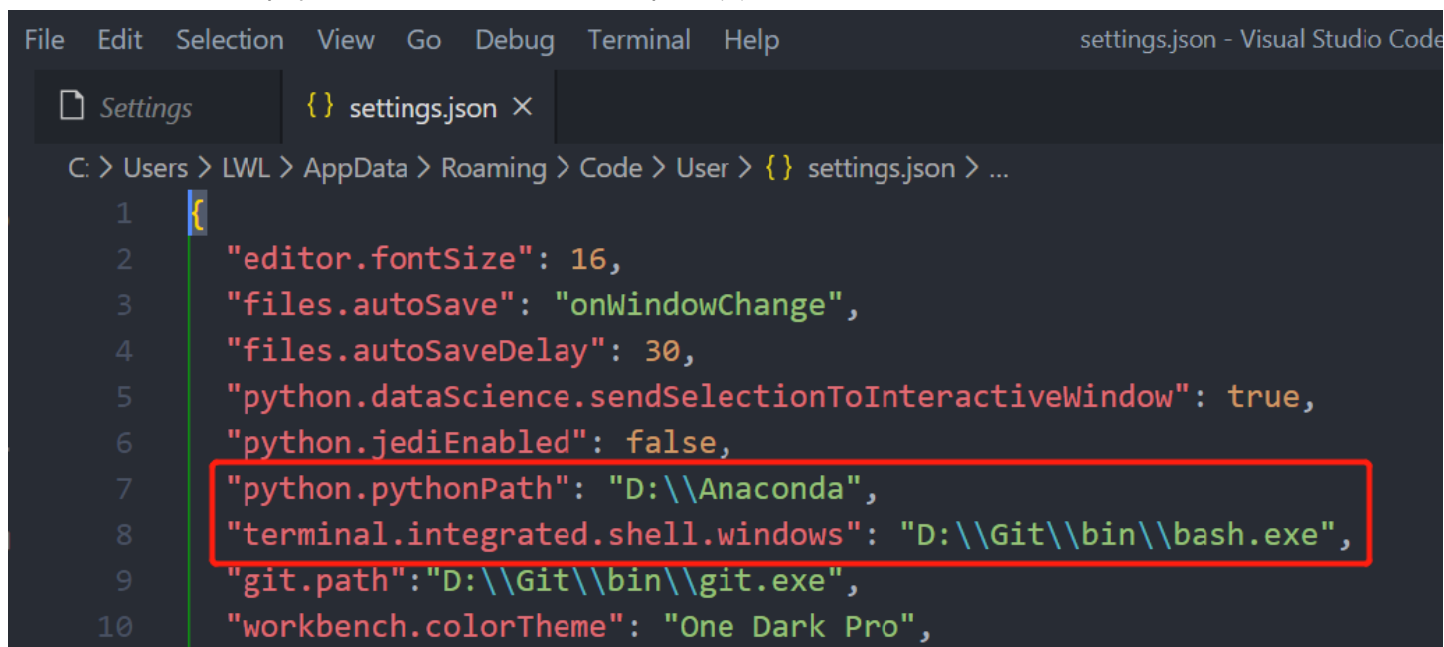


3). settings json添加配置

添加内容为：

```
"terminal.integrated.shell.windows": "D:\\Git\\bin\\bash.exe",  
"git.path": "D:\\Git\\bin\\git.exe"
```

注意后面的文件路径，要根据自己的电脑上的路径来填写



4). 添加完成之后，按 Ctrl + S 保存 settings json，再重启 VSCode 就可以在 VSCode 使用 Git 工具。



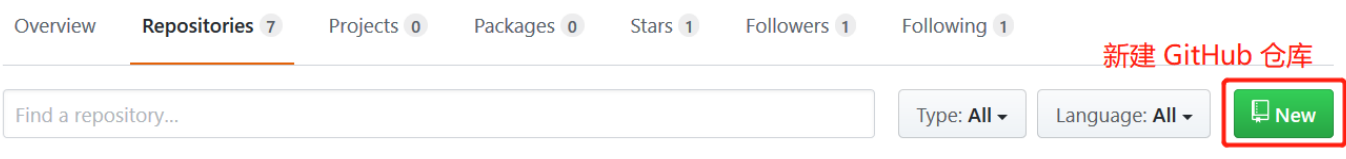
- 到此准备工作全部做完，下一步就是对计算器项目进行演示。

2. 范例演示

- 此次演示是为了让读者了解如何在项目中使用 Git 工具加入版本控制功能以及如何将本地项目存储到 GitHub 仓库。
- 此次演示分 4 个步骤进行：
 1. 在 GitHub 上建立 calculator 仓库
 2. 将 GitHub 上的 calculator 仓库的内容克隆到本地
 3. 将计算器项目里的内容添加到 Git 仓库
 4. 将 Git 仓库的内容推送到 GitHub 的 calculator 仓库

(1) 第一步，在 GitHub 上建立 calculator 仓库

- 在 GitHub 上登录你的账号，进入 Your repositories 界面，点击 New，进入创建 GitHub 仓库界面



- 填写相应信息，然后点击 Create repository，就可以创建一个 GitHub 仓库

Owner

WanglinLi595

Repository name *

calculation

项目名

Great repository names are short and memorable. Need inspiration? How about **effective-dollop**?

Description (optional)

这是一个用 python 语言写的计算器项目

项目描述

☒ Public

Anyone can see this repository. You choose who can commit.

☐ Private

You choose who can see and commit to this repository.

选择仓库类型，为公有仓库还是私有仓库

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

☒ Initialize this repository with a README

是否初初始化 README 文件

This will let you immediately clone the repository to your computer.

Add .gitignore: Python

Add a license: MIT License

是否初始化 .gitignore

选择项目许可证

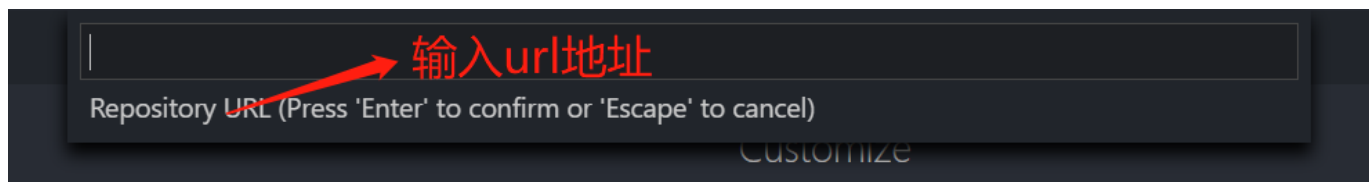
Create repository

创建仓库

- 此时的 GitHub 仓库只有几个初始化的文件，需要我们将计算器项目的文件推送到 GitHub 仓库里

(2) 第二步，将 GitHub 上的 calculator 仓库的内容克隆到本地

- GitHub 上的 calculator 仓库创建完成之后，打开 VSCode，按下 Shift + Ctrl + p，进入命令搜索框，在里面输入 Git Clone，然后点击 Git Clone 命令，再输入 calculator 仓库的 url 地址。



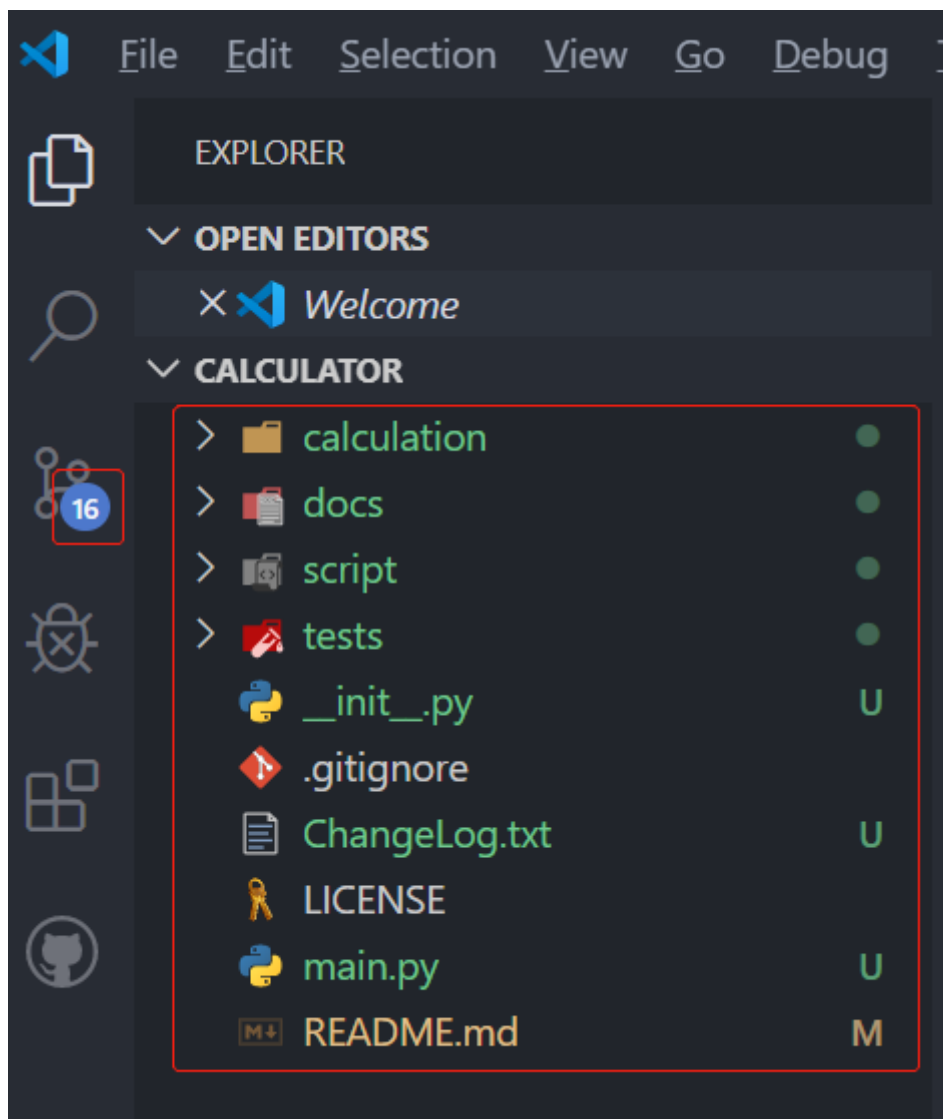
- 按下换行键，会弹出一个窗口，这是用来选择 GitHub 仓库克隆到本地的路径。选择好之后，点击 Select Repository Location 就可以将 calculator 仓库的内容克隆到本地。

(3) 第三步，将计算器项目里的内容添加到 Git 仓库

- calculator 仓库克隆到本地后，会在你选定的路径生成一个文件夹。文件夹的名字为你 GitHub 仓库的名字。我们 GitHub 上的仓库名为 calculator，那么克隆下来的文件夹名也为 calculator。

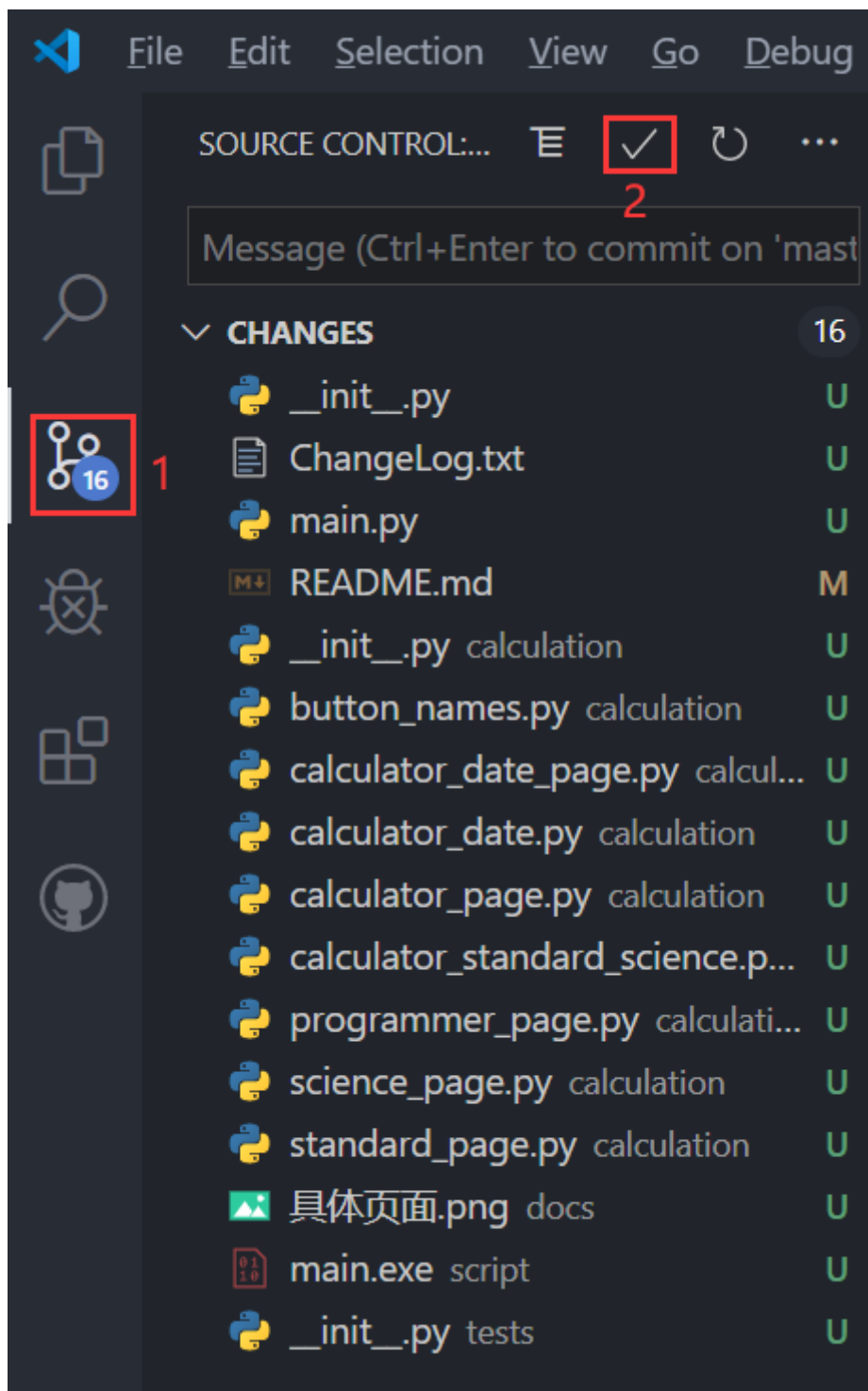
名称	修改日期	类型
pythonProject	2019/10/23 20:01	文件夹
PyQt5_Graphics_View	2019/12/11 19:16	文件夹
OpeoCV_Image_Process_Study	2019/12/16 19:08	文件夹
opencvProject	2019/10/17 19:11	文件夹
OpenCV_Function_Demonstration	2019/12/23 14:43	文件夹
OpenCV	2019/12/19 15:54	文件夹
modStopWatch	2019/10/25 10:44	文件夹
Kugou	2019/12/14 20:03	文件夹
image_process_V1_2	2019/12/11 14:45	文件夹
image_process_V1_1	2019/12/10 8:06	文件夹
image_process_V1_0	2019/12/10 8:06	文件夹
GitHub	2019/11/14 19:58	文件夹
example	2019/12/20 18:35	文件夹
calculator	2019/12/25 14:55	文件夹
32 Project	2019/10/13 21:08	文件夹
.ipynb checkpoints	2019/12/18 11:34	文件夹

- 然后将计算器项目里面的内容复制到 calculator 文件夹里面。用 VSCode 打开 calculator 文件夹。



从图中可以看到，VSCode 已经识别到了我们修改的文件

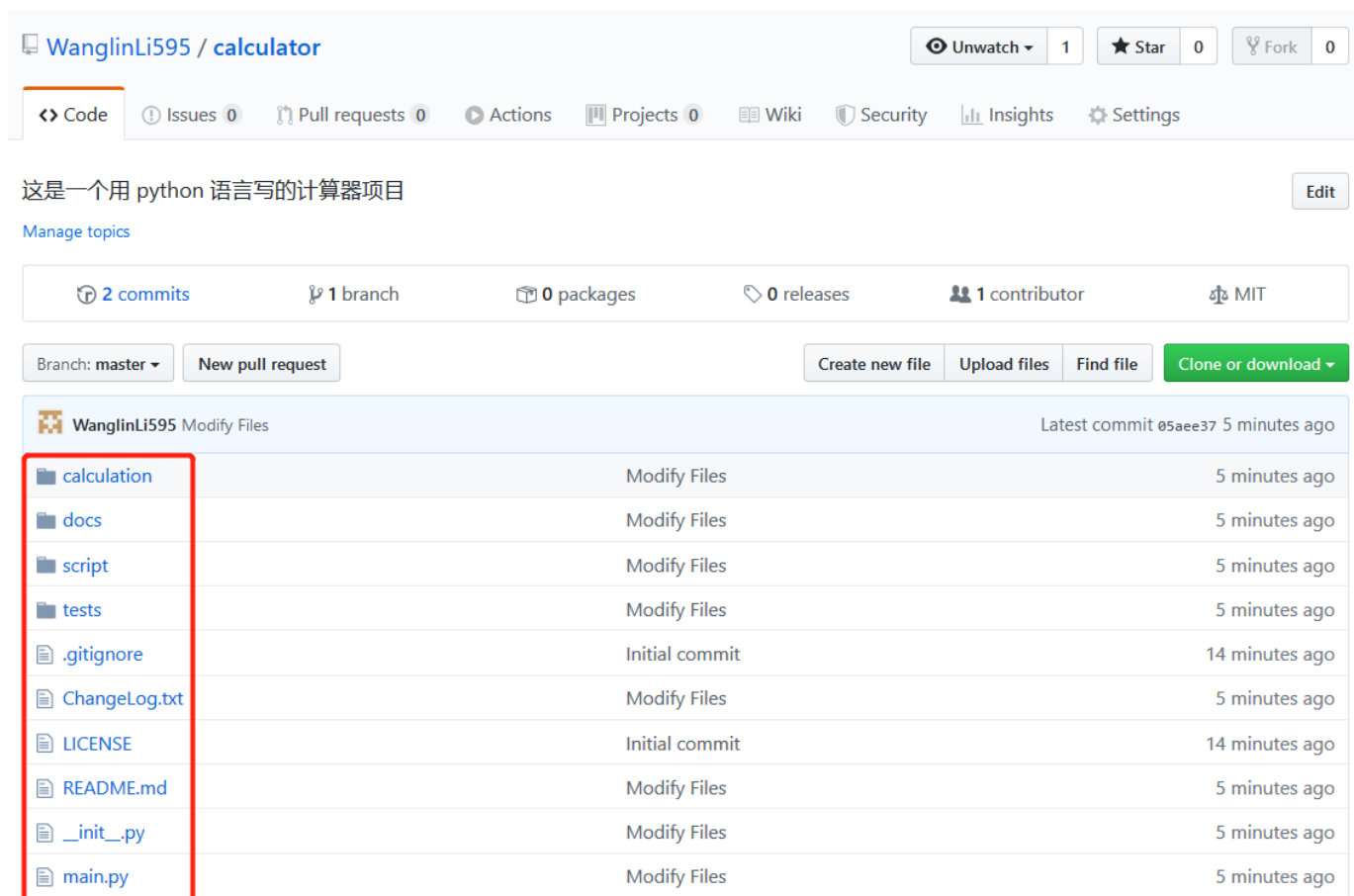
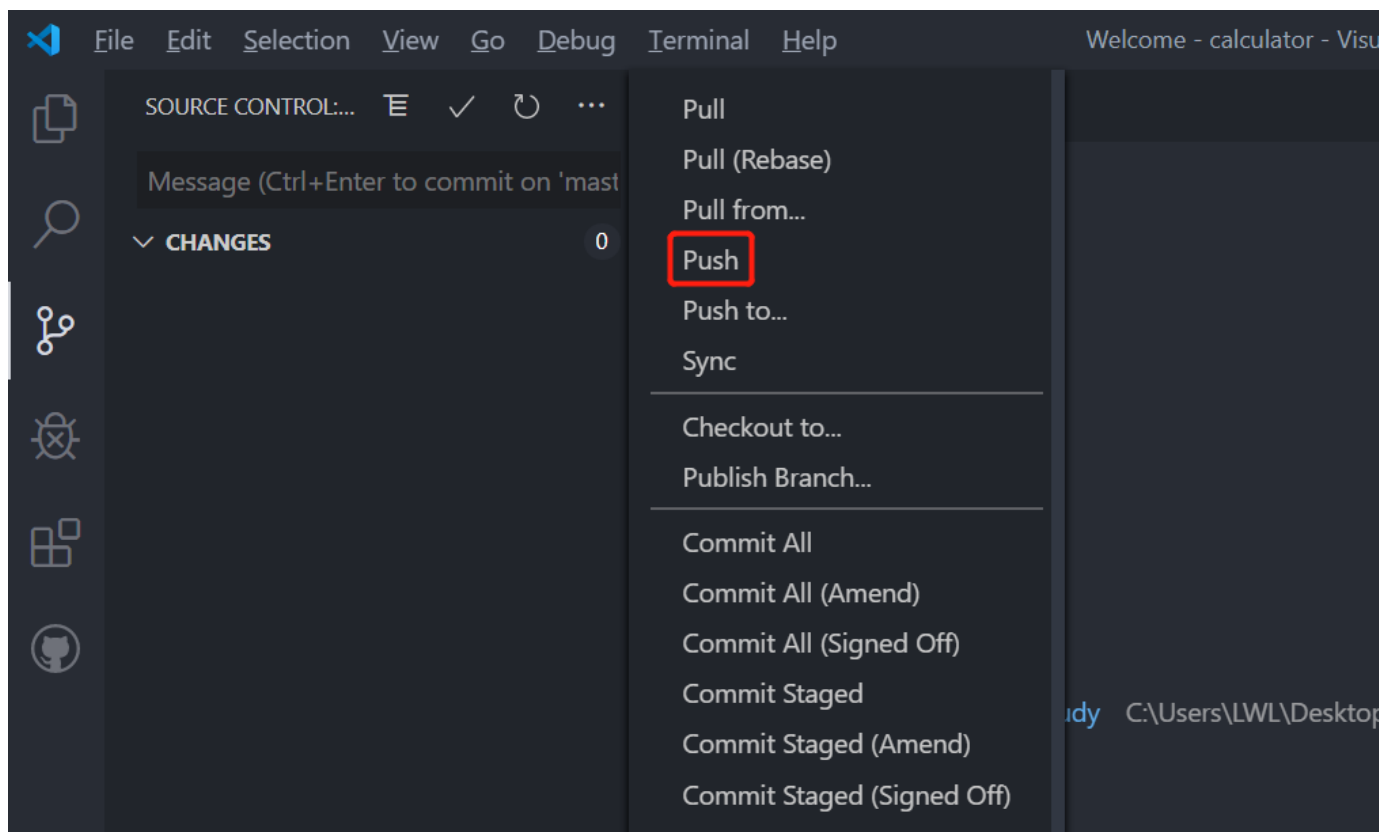
- 点击 Source Control，再点击 "√" (这个相当于 `git commit -am` 命令)，它会将所有修改的文件添加到 Git 仓库。



- 点击 "✓" 之后会出现一个警告消息框，点击 Yes 即可。之后又会出现一个消息填写框，在里面输入修改信息即可。这样就将计算器项目里的内容添加到 Git 仓库了。
- 此时该项目就能进行版本控制了

(4) 第四步，将 Git 仓库的内容推送到 GitHub 的 calculator 仓库

- 将修改文件添加到 Git 仓库后，只需在 VSCode Git 工具栏点击 push，就可以将 Git 仓库里面的内容推送到 GitHub 的 calculator 仓库了。



可以看到，此时本地计算器项目文件夹里的文件已经存储到了 GitHub 上的 calculator 仓库。

- 经过这四个步骤，计算器项目就可以使用 Git 进行版本控制，而且计算器项目也存储在 GitHub 仓库了。
- 这里只演示了在 VSCode 上使用 Git 进行版本控制的基本步骤。如果了解 Git 进行版本控制的更多功能，如：版本回退，分支管理，团队协作等，请查看下面的推荐教程。

四. 更多教程推荐

- (1) [Git 官网](#)
- (2) [Git 教程 | 菜鸟教程](#)
- (3) [Git教程 - 廖雪峰的官方网站](#)