|  |  |  |
| --- | --- | --- |
|  |  | **分级B** |
|  | **Git&Github使用流程** |  |
|  | 敏捷动力团队 |  |

|  |  |  |
| --- | --- | --- |
| **编写** | 郑永博 | 2023-05-22 |
| **修订** | 蔡泽凯 | 2023-05-25 |
| **审阅** |  |  |

|  |  |
| --- | --- |
| **分级** | 🞎 A. 内部申请获取  🗹 B. 内部自由获取  🞎 C. 外部自由获取 |

**目录**

[1. Git下载和安装 1](#_Toc6692)

[1.1. Git下载 1](#_Toc2055)

[1.2. Git安装 1](#_Toc527)

[2. Git&Github快速入门 1](#_Toc31059)

[2.1. Git&Github基本配置 1](#_Toc4643)

[2.2. Git&Github基本用法 4](#_Toc32718)

[3. Git常见用法介绍 9](#_Toc12969)

[3.1. 查看文件状态（git status） 9](#_Toc31215)

[3.2. 保存未提交修改（git stash） 10](#_Toc19021)

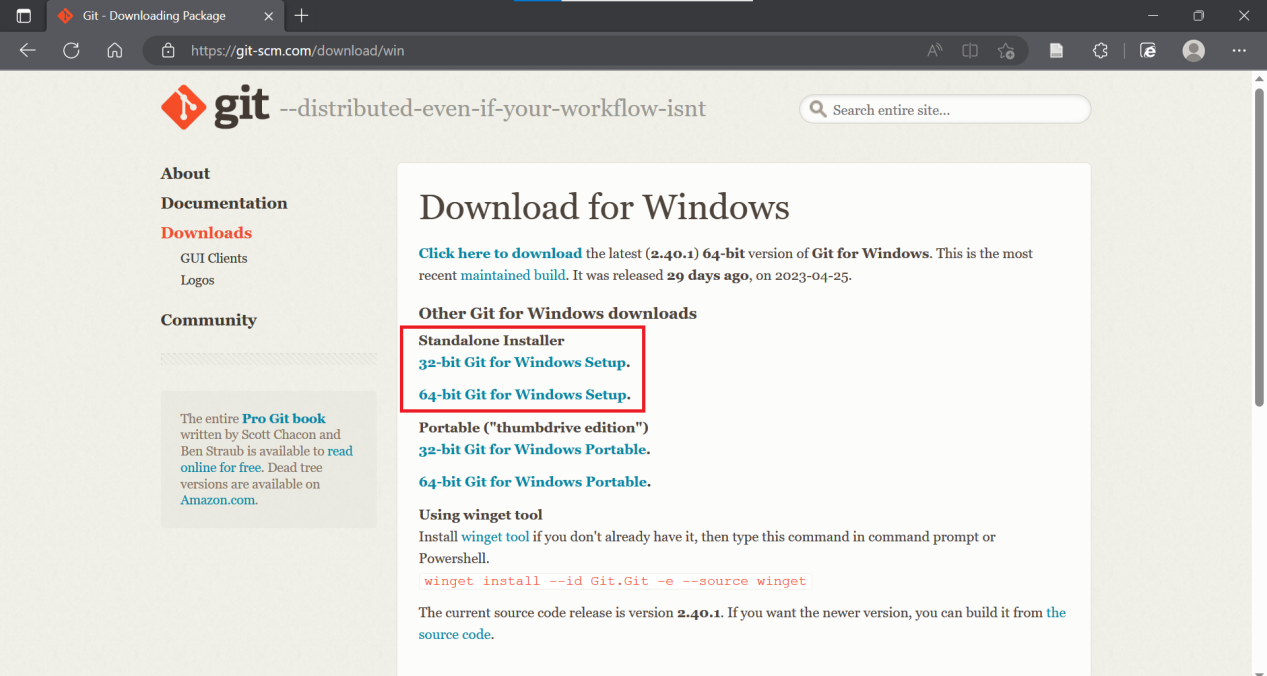
[3.3. 回退到某一次提交（git reset） 11](#_Toc21908)

[3.4. 配置.gitignore文件 12](#_Toc12743)

[3.5. 本地仓库新建与远程分支关联的分支 13](#_Toc2442)

1. Git下载和安装
   1. Git下载

下载链接：https://git-scm.com/download/win。根据电脑是32位操作系统还是64位操作系统下载对应的Git版本，如图1。



1. Git下载
   1. Git安装

双击下载的Git安装包，除了选择安装路径之外，其他直接点击“Next”直到安装完成即可。

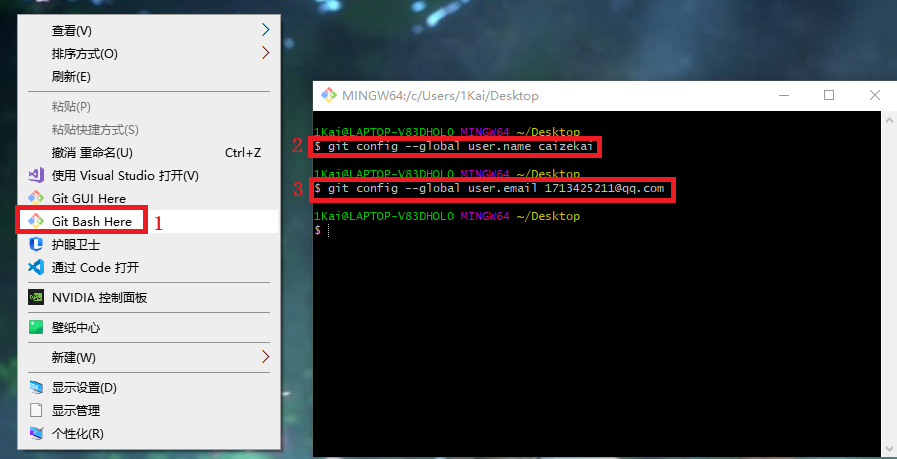
1. Git&Github快速入门
   1. Git&Github基本配置

这些基本配置是针对Git&Github新用户而言，如果之前设置过这些操作可以直接跳过。

1. Git设置用户

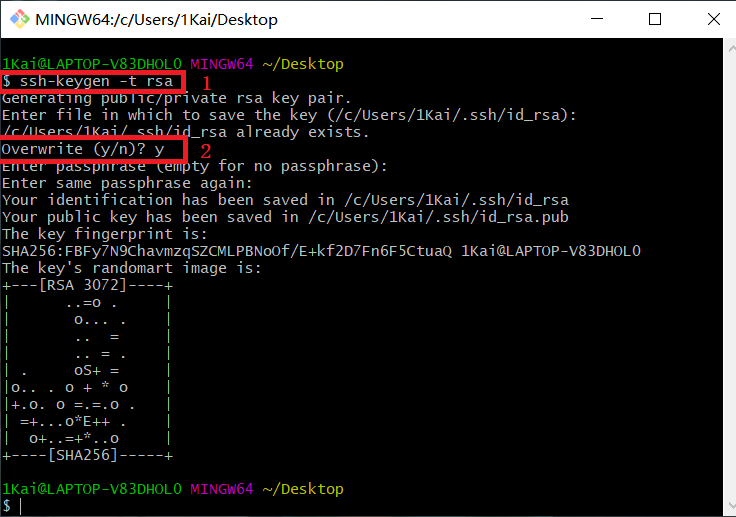
电脑第一次使用Git需要设置用户信息，如图2，桌面右击，点击“Git Bash Here”，在命令行窗口设置Git用户信息：“git config --global user.name Your Name”，“git config --global user.email email@example.com”。

1. 本文可能会某些文字使用蓝色或者橙色进行标记，这些文字是git命令，蓝色部分是不需要用户更改，橙色部分需要用户依据具体情况更改。



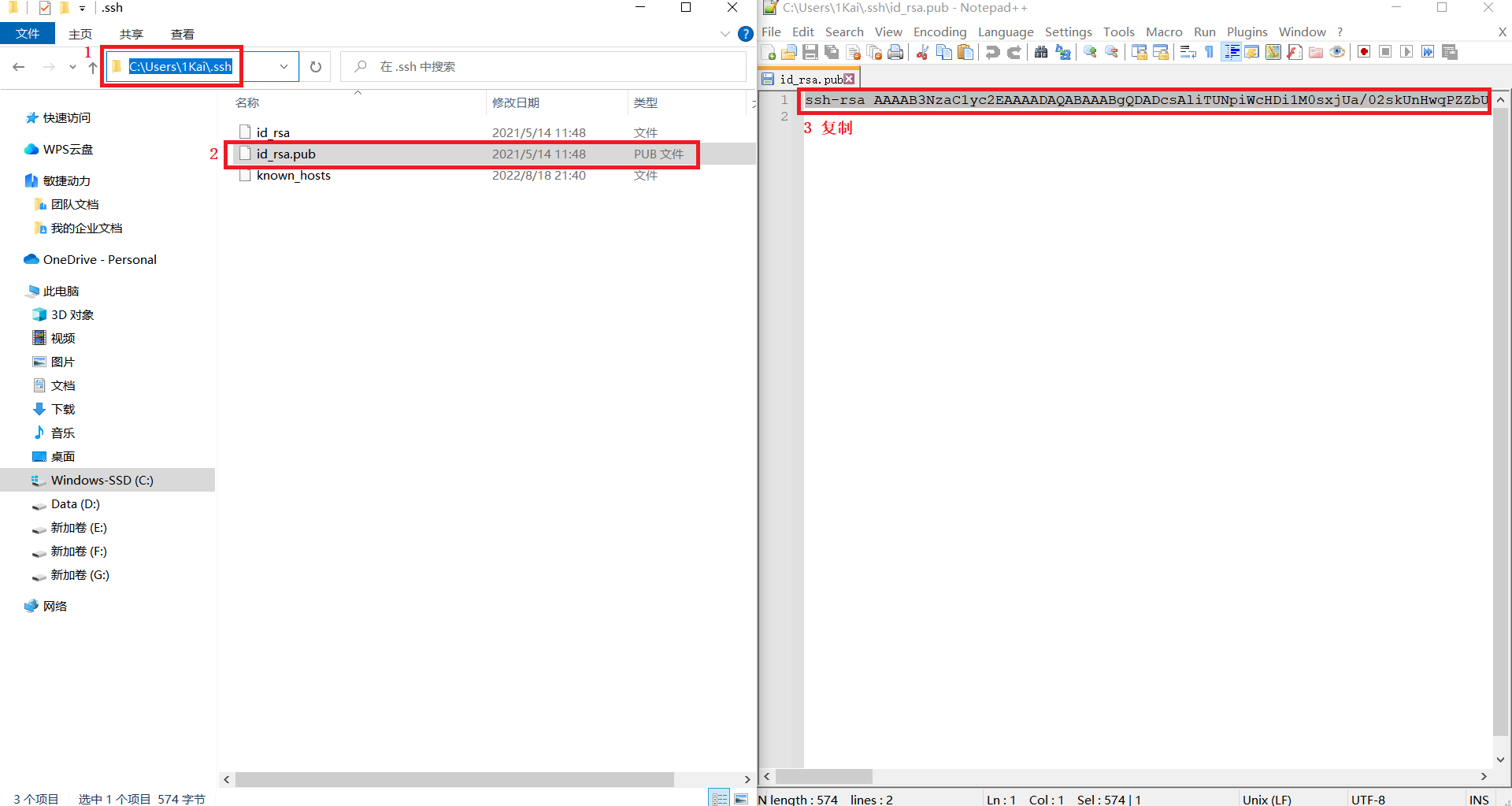
1. Git设置用户信息
2. 添加SSH Keys到Github账户

首先如图3，打开“Git Bash”，输入命令“ssh-keygen -t rsa”生成SSH密钥，然后一直回车即可。



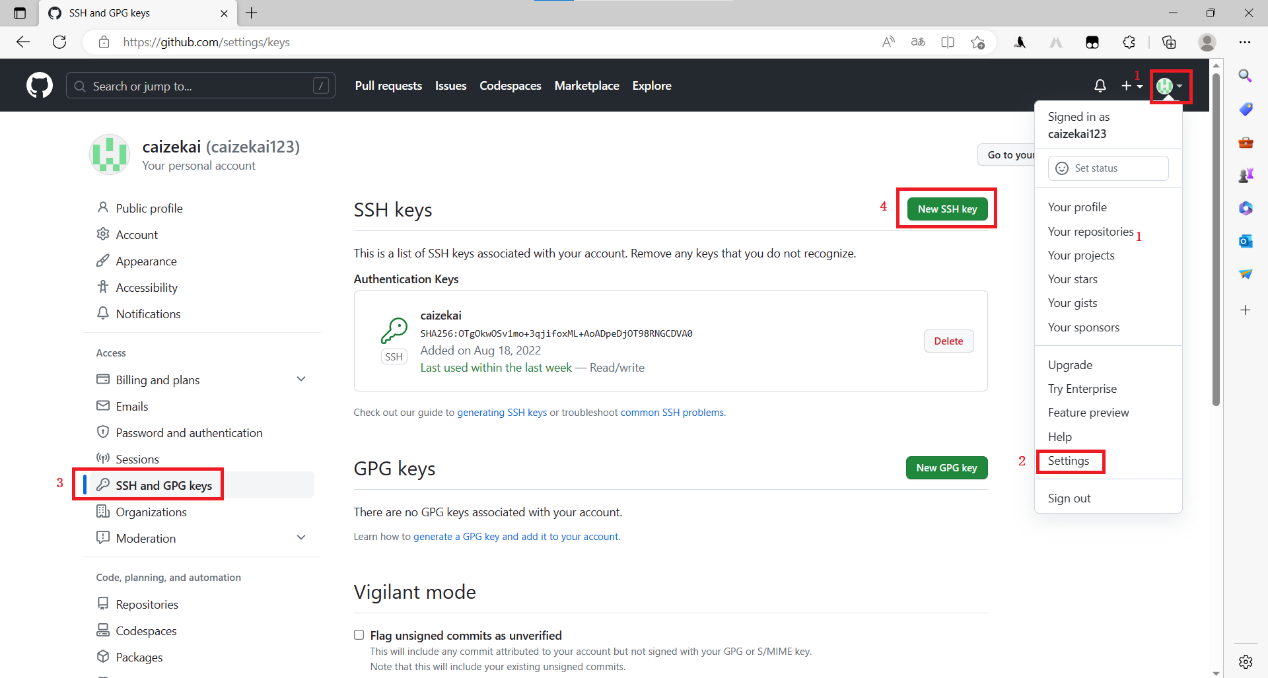
1. SSH 密钥生成
2. 如果之前生成过SSH密钥，会提示是否重写，直接填“n”即可，此处是为演示填写了“y”。

接着如图4，在目录“C:\Users\XXX\.ssh\”（XXX是用户名）下打开“id\_rsa.pub”，复制公钥。



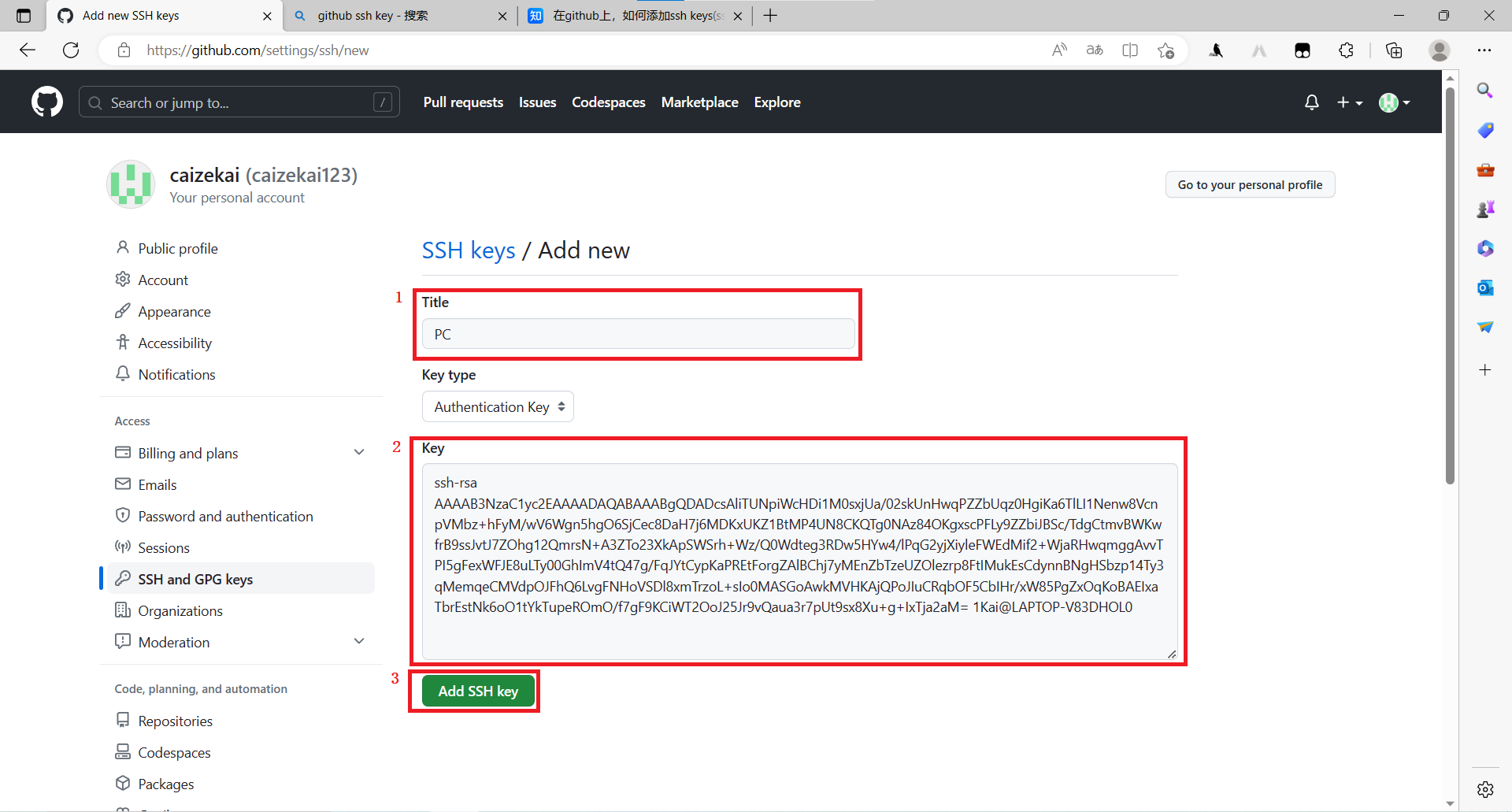
1. 复制公钥

接着如图5，在Github账号上依次点击“头像”，点击“Settings”，点击“SSH and GPG keys”，点击“New SSH key”。



1. 添加SSH Key 1

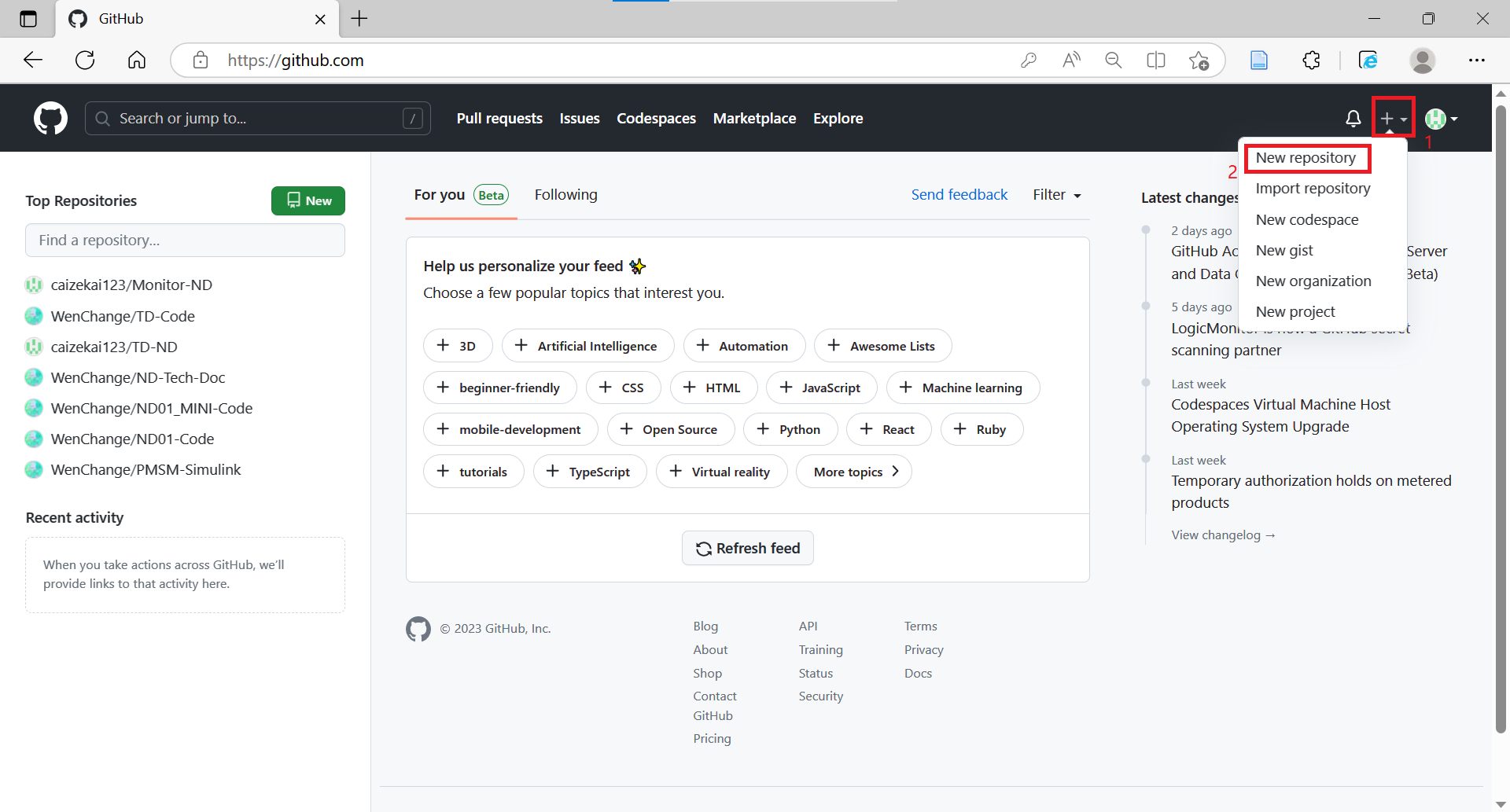
最后如图6，依次填写“Title”（描述当前SSH Key来自哪个设备）、粘贴“Key”，点击“Add SSH key”。



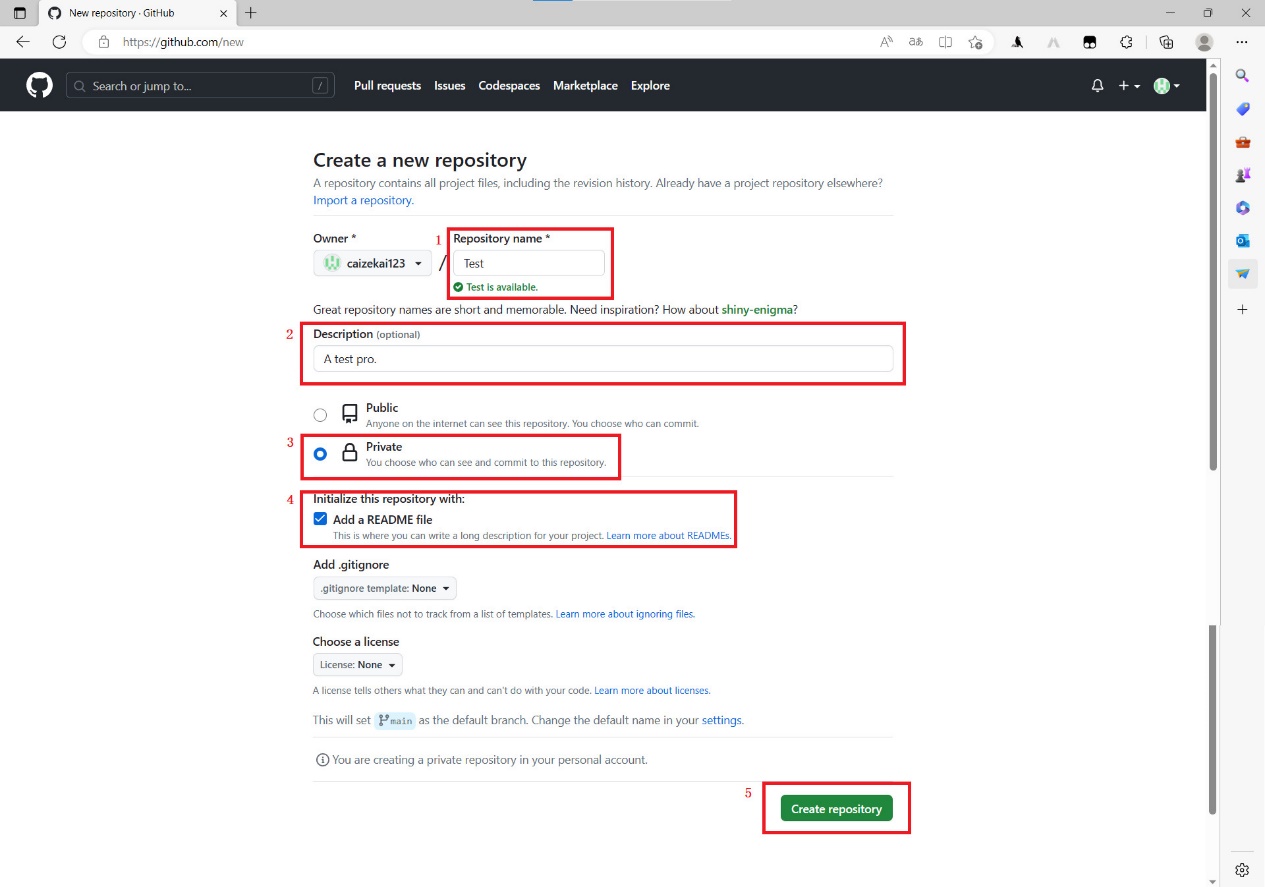
1. 添加SSH Key 2
   1. Git&Github基本用法
2. 如果无需在Github上创建仓库可直接阅读S5-S7。

访问Github网站https://github.com/，登录Github账号。按照如下步骤进行操作：

S1：创建私有仓库。首先如图7，点击右上角“+”，在下拉框点击“New repository”；接着如图8，依次填写仓库名称“Repository name”，仓库描述（可选）“Description”，选择私有仓库“Private”，勾选“Add a README file”，其他不用管，最后点击“Create repository”。



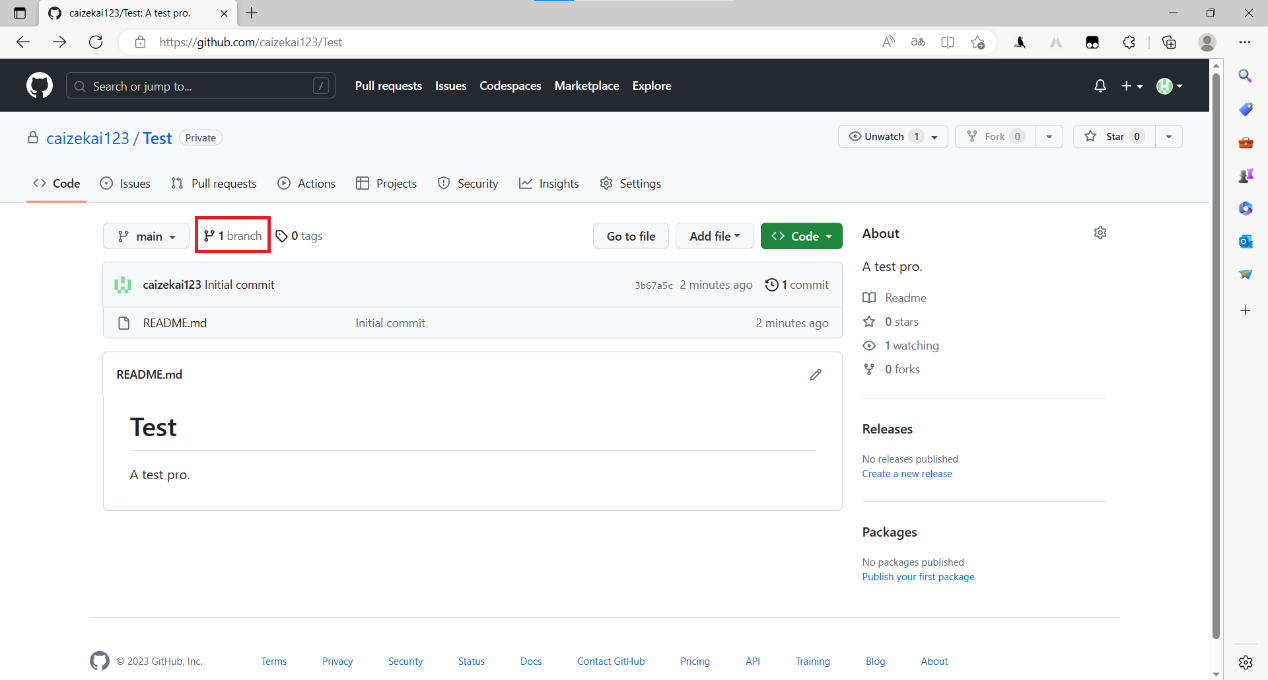
1. S1：创建私有远程仓库1



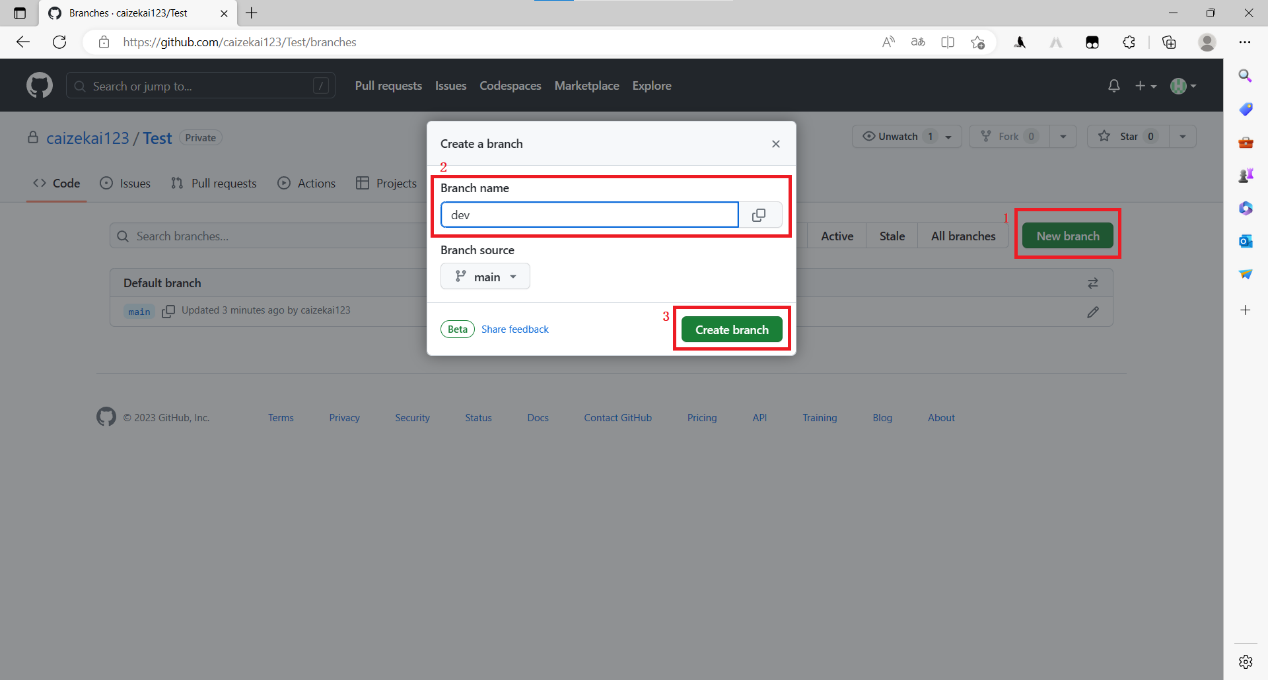
1. S1：创建私有远程仓库2

S2：创建开发分支（dev）。首先如图9，点击“1 branches”；接着如图10，依次点击“New branch”，填写分支名称“Branch name”，点击“Create branch”。

1. 一般来说，Git管理的项目一般不会在主分支（master/main）进行开发，而是会创建开发分支（dev），在dev上开发，当dev的开发有阶段性成果时，再将dev分支合并到主分支上，从而确保主分支（master/main）的代码是相对稳定的。

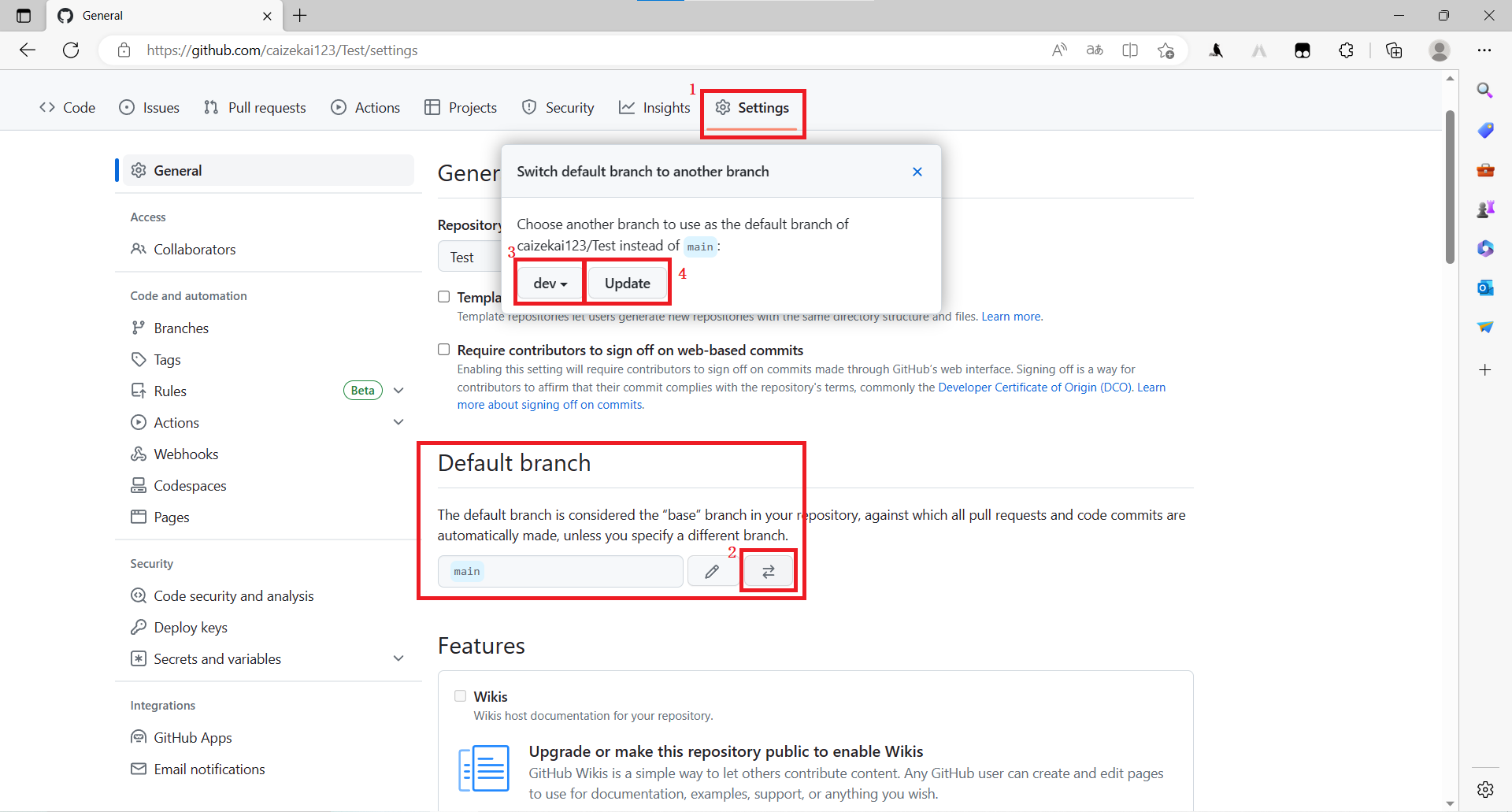


1. S2：创建开发分支（dev）1



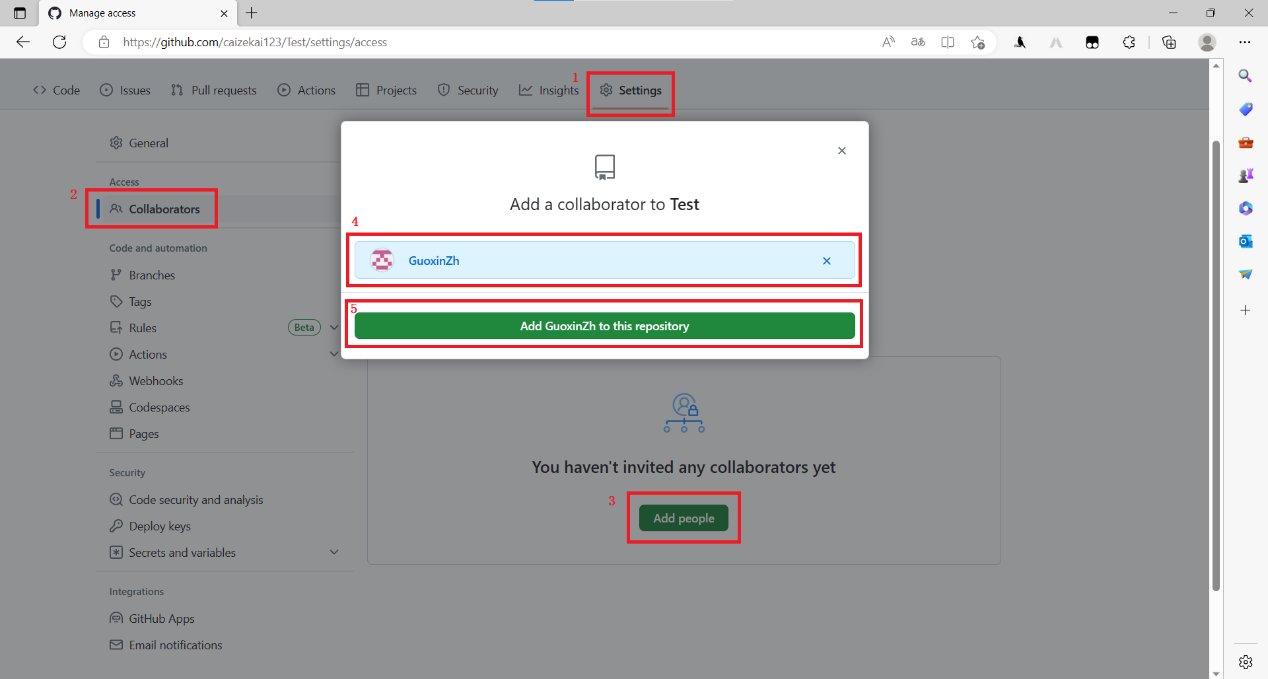
1. S2：创建开发分支（dev）2

S3：修改默认分支为dev。如图11，依次点击“Settings”，在“Default branch”点击“左右箭头”，选择要切换的分支“dev”，点击“Update”。



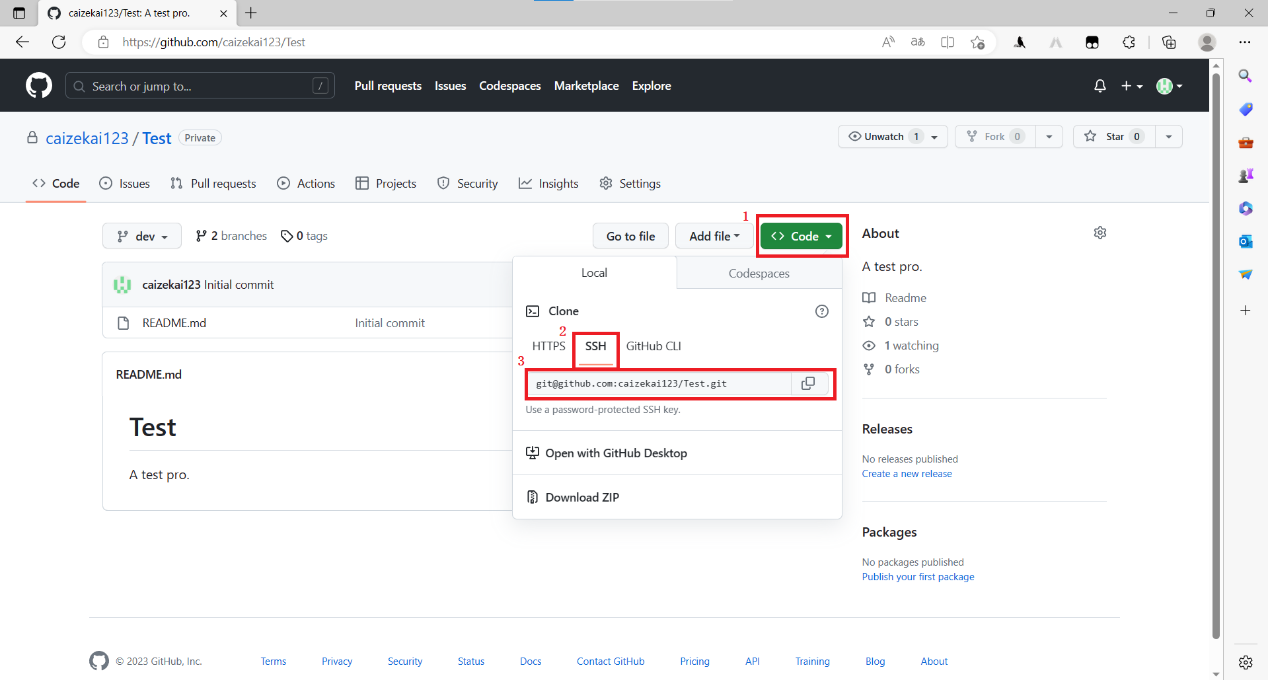
1. S3：修改默认分支为dev

S4：添加合作者。如图12，依次点击“Settings”，点击“Collaborators”，点击“Add people”，填写用户名搜索需要添加的合作者，点击“Add xxx to this repository”。

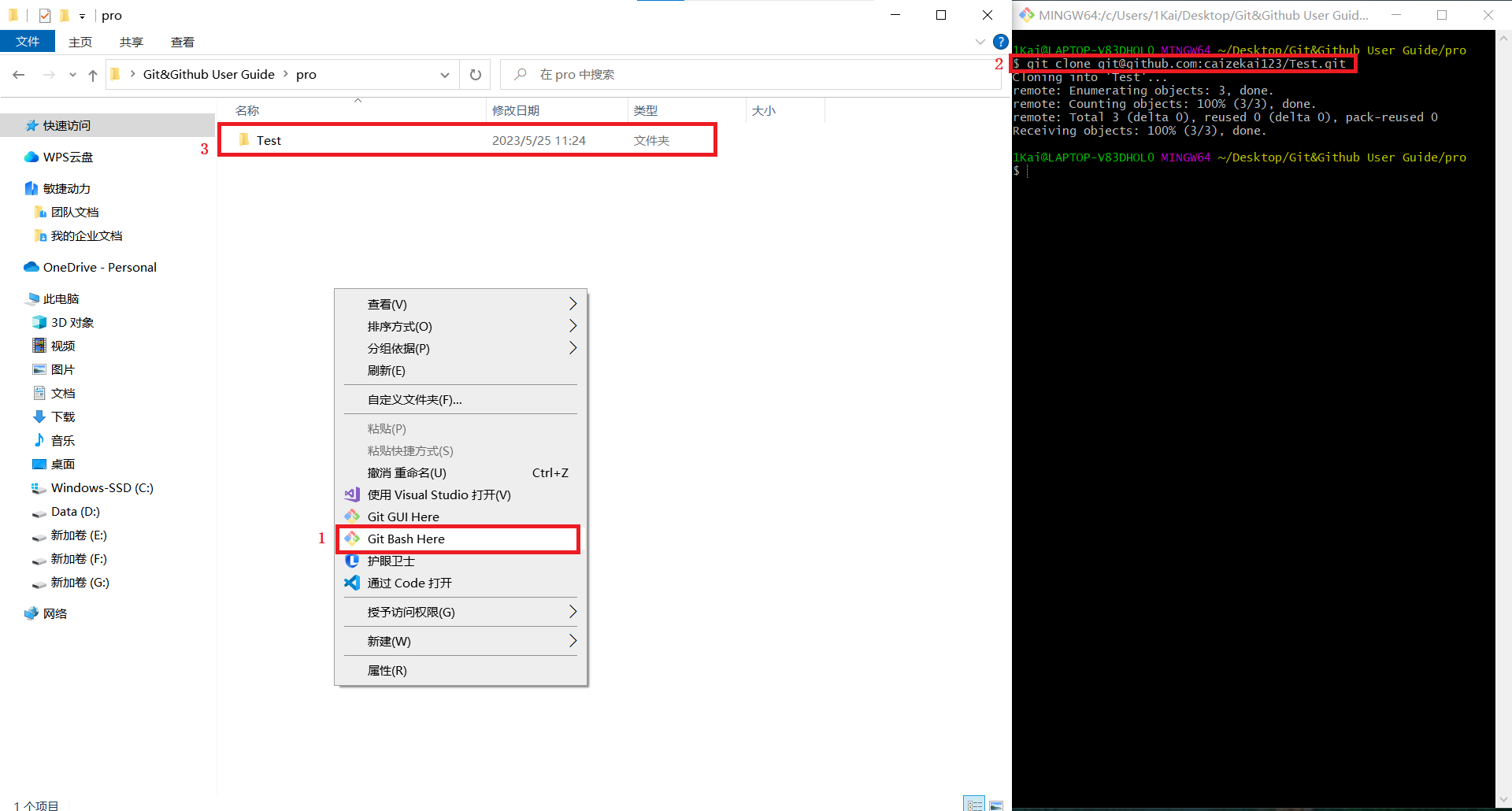


1. S4：添加合作者

S5：克隆远程仓库到本地仓库。首先如图13，依次点击“Code”，点击“SSH”，复制“网址”；接着如图14，在电脑的某个文件夹里右击，然后点击“Git Bash Here”，在弹出的命令行窗口，输入“git clone 网址”，克隆成功后会文件夹会出现克隆的工程。

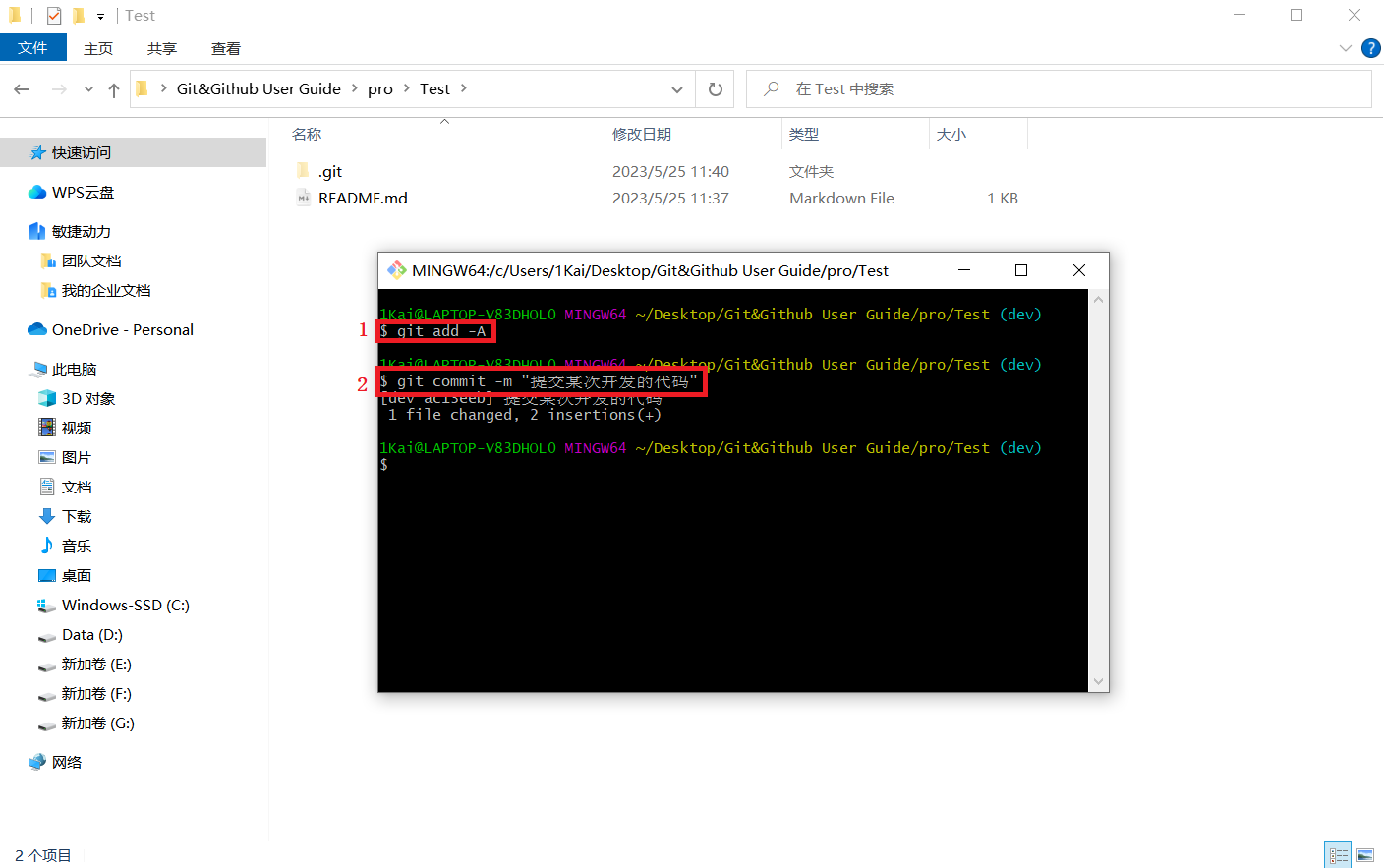


1. S5：克隆远程仓库到本地仓库1



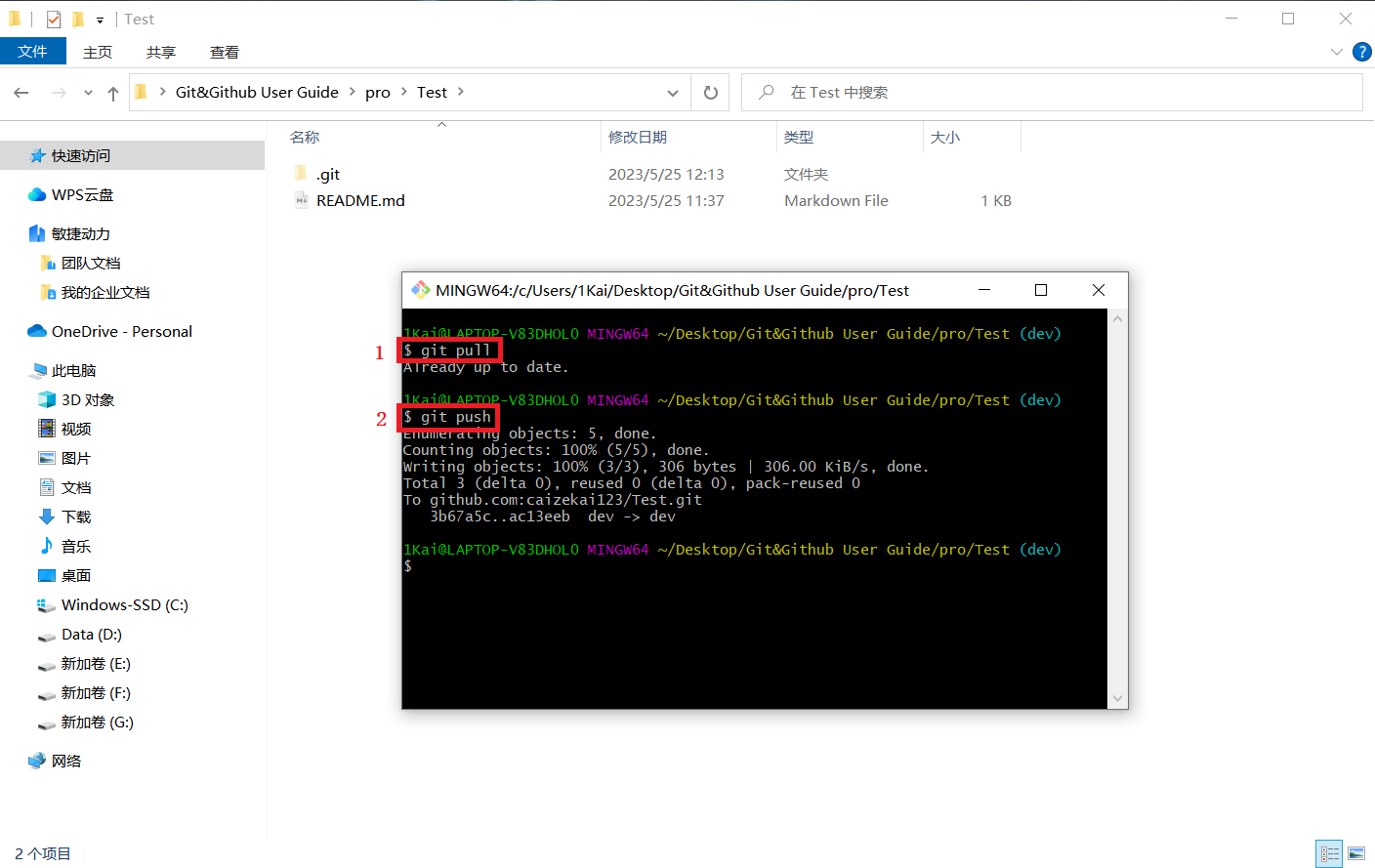
1. S5：隆远程仓库到本地仓库2

S6：本地仓库提交代码。在完成某次开发后，需要将此次开发的代码提交的本地仓库。具体操作如图15，在工程所在文件夹中，右击打开Git Bash，在命令行输入“git add -A”回车，再输入“git commit -m “填写本次提交的说明””。



1. S6：本地仓库提交代码

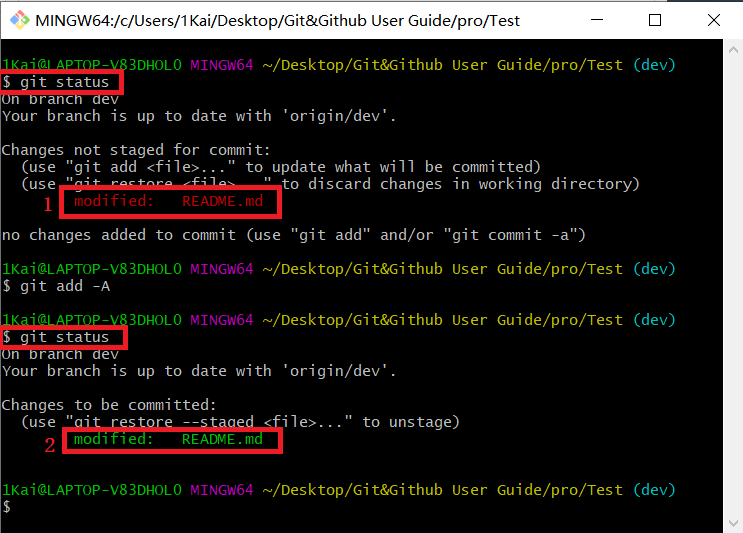
S7：本地仓库推送提交到远程仓库。如图16，先拉取远程仓库“git pull”，再将本地仓库推送提交到远程仓库“git push”。



1. S7：本地仓库推送提交到远程仓库
2. 本地仓库推送提交到远程仓库需要先拉取的原因是远程仓库可能有其他合作者提交了代码，但这些代码并没有合并到当前的本地工程中，因此需要先拉取远程仓库再将此次提交推送到远程仓库。
3. Git常见用法介绍
   1. 查看文件状态（git status）

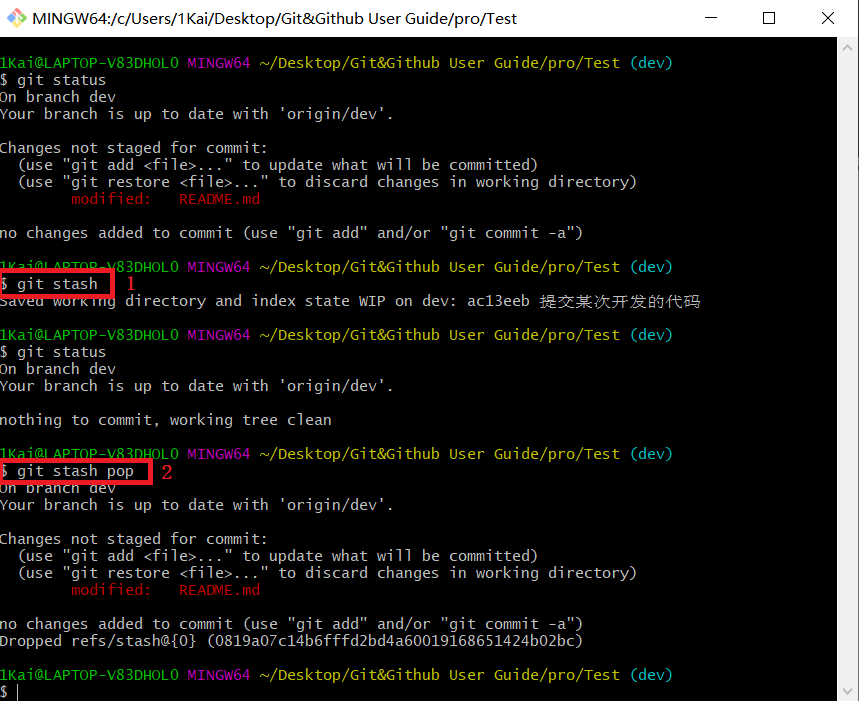
“git status”用来查看文件状态，如图17，图中红色字样“modified: README.md”表示这些文件进行了修改，但是没有被本地仓库记录，处于工作区，使用“git add -A”后，红色字样变成绿色字样“modified: README.md”，表示文件被暂存到暂存区，此时文件已经被本地仓库记录，但还没有提交到当前分支。

1. 关于工作区、暂存区、本地仓库、远程仓库的概念有需要请自行了解。



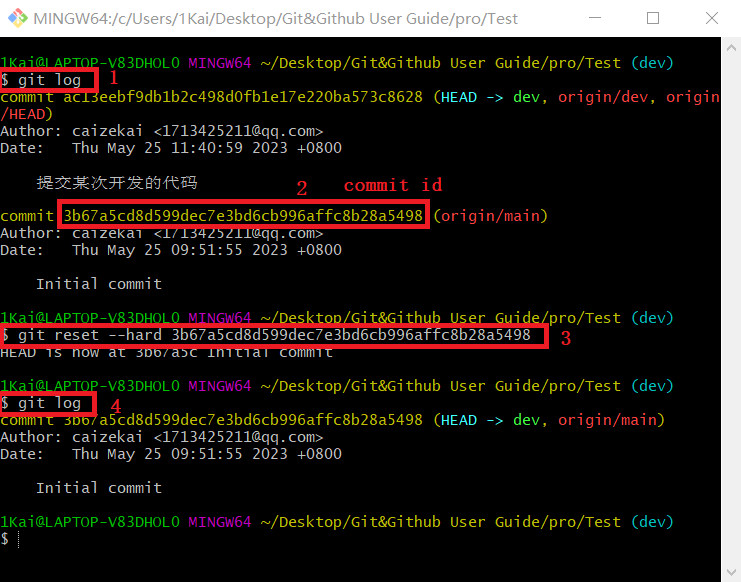
1. 查看文件状态
   1. 保存未提交修改（git stash）

在某些情况下，可能需要将当前开发的未提交的代码保存起来，然后进行版本回退或者远程拉取的操作。如图18，使用“git stash”将当前工作区和暂存区的内容进行保存，如果保存的内容需要恢复到工作区和暂存区使用“git stash pop”。



1. 保存未提交修改
   1. 回退到某一次提交（git reset）

在某些情况下，可能需要回退到某一次提交去查看之前的代码。如图19，首先使用“git log”查看当前仓库的所有提交，复制需要回退的“commit id”，使用“git reset --hard commit id”回退。



1. 回退到某一次提交
   1. 配置.gitignore文件

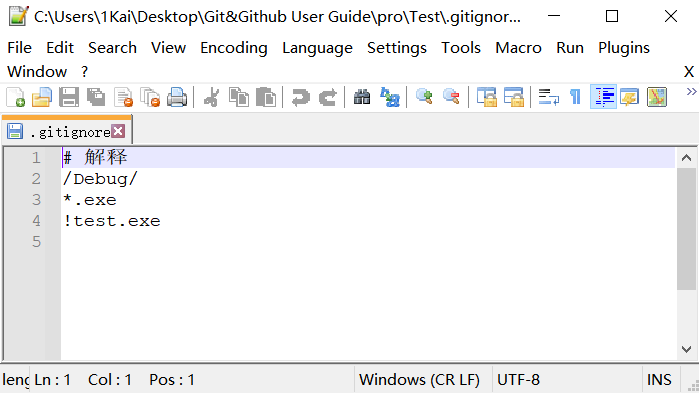
对于某个代码工程，有些文件并不需要被git追踪（例如C语言代码工程编译过程产生的.o文件等）。因此可以在仓库目录下配置“.gitignore”文件（文件名直接命名为“.gitignore”即可）。“.gitignore”文件的编写主要介绍4点：

（1）#：表示注释；

（2）文件夹忽略：例如“/Debug/”表示不追踪当前路径下的Debug文件夹里的文件；

（3）特定后缀名文件忽略：例如“\*.exe”表示不追踪当前路径下的所有“.exe”后缀的文件;

（4）使用“!”不忽略特定文件：例如“!test.exe”表示“test.exe”不忽略，常见用法是在使用“\*.exe”后会忽略exe类型文件，此时再使用“！test.exe”则test.exe不会被忽略，而其他的exe类型文件仍被忽略。



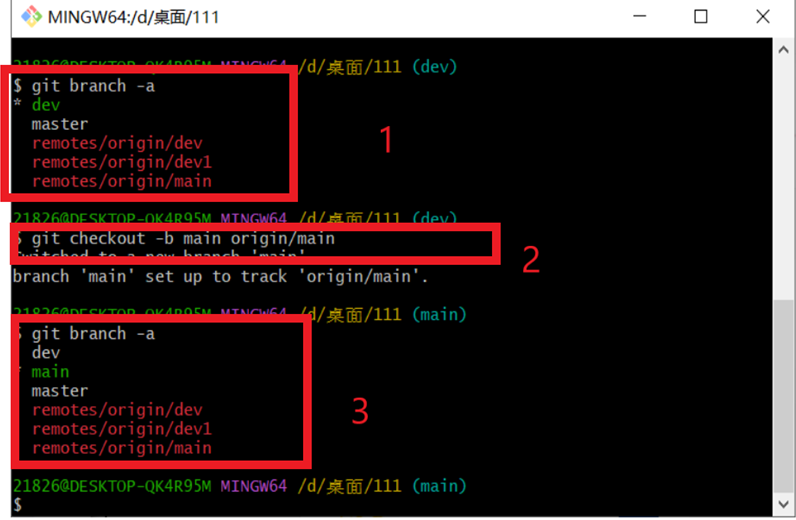
1. .gitignore文件编写
   1. 本地仓库新建与远程分支关联的分支

在某些情况下，需要在本地仓库新建与远程分支关联的分支。可按以下步骤进行（图21）：

1.首先使用“git branch -a”查看所有分支列表，列出的分支列表名称会用“\*”标记当前分支，红色字体表示远程分支。

2.使用“git checkout -b 本地分支名 origin/远程分支名”，在本地仓库新建与远程分支关联的分支。

注7：在分支创建时，“本地分支名”和“远程分支名”要相同，否则在使用“git push”时报错：“fatal: The upstream branch of your current branch does not match the name of your current branch”。



1. 本地仓库新建与远程分支关联的分支