

Git基础知识

陈加忠

计算科学与技术学院

QQ群: 904987289

Email: jzchen@hust.edu.cn

<http://cshust.gitee.io/>

<https://gitee.com/chenjz70>

版本控制

- 在实验室里实验做了一半, 怎么保存代码?
- 对于一个大项目, 版本控制可帮助开发人员跟踪和管理软件项目代码的更改
- 如果核心开发人员想在代码库的某个特定部分上工作, 那么直接编辑“官方”源代码是不安全或无效的, 而版本控制允许开发人员安全地完成分支和合并
- 通过分支, 开发人员复制部分源代码并可安全更改代码的这一部分, 而不会影响项目的其余部分
- 而且, 一旦开发人员所写的部分代码正常工作, 他就可以将该代码合并回主要源代码以使其正式化
- 然后跟踪所有这些更改, 并在需要时还原. 相当于做backup, 以免自己忘记改了代码的哪些地方

版本控制的历史

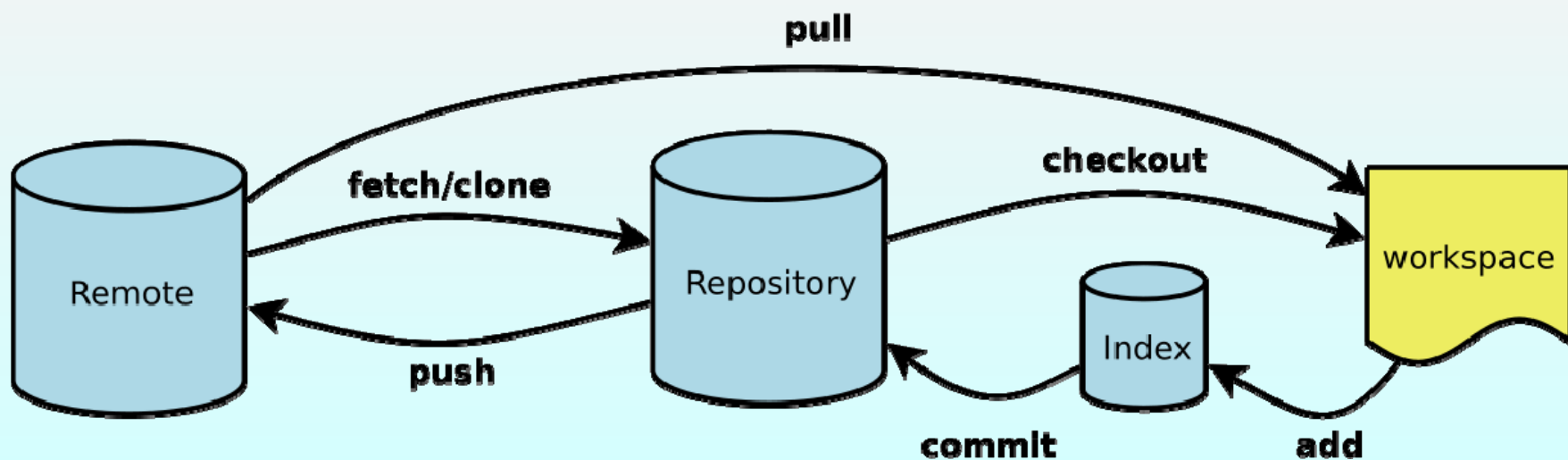
- 手工管理
- 日志记录
- 集中式版本控制 (cvs, svn)
- 分布式版本控制 (Git)

Gitee和Github

- Git是一个分布式版本控制系统,这意味着每个开发人员的计算机上都可以使用整个代码库和历史记录,这样可以轻松进行分支和合并
- GitHub是一家商业公司,提供基于云的Git存储库托管服务
- Gitee: Gitee和Github是同一类,在云端
 - ✓ 区别是Github是国外的, Gitee是国内的,二者的使用均需要借助Git, **Gitee访问速度比Github快!!!**
 - ✓ Gitee是oschina.net推出的代码托管平台,支持Git,提供**免费的私有仓库托管**,目前已有超过350万的开发者选择码云

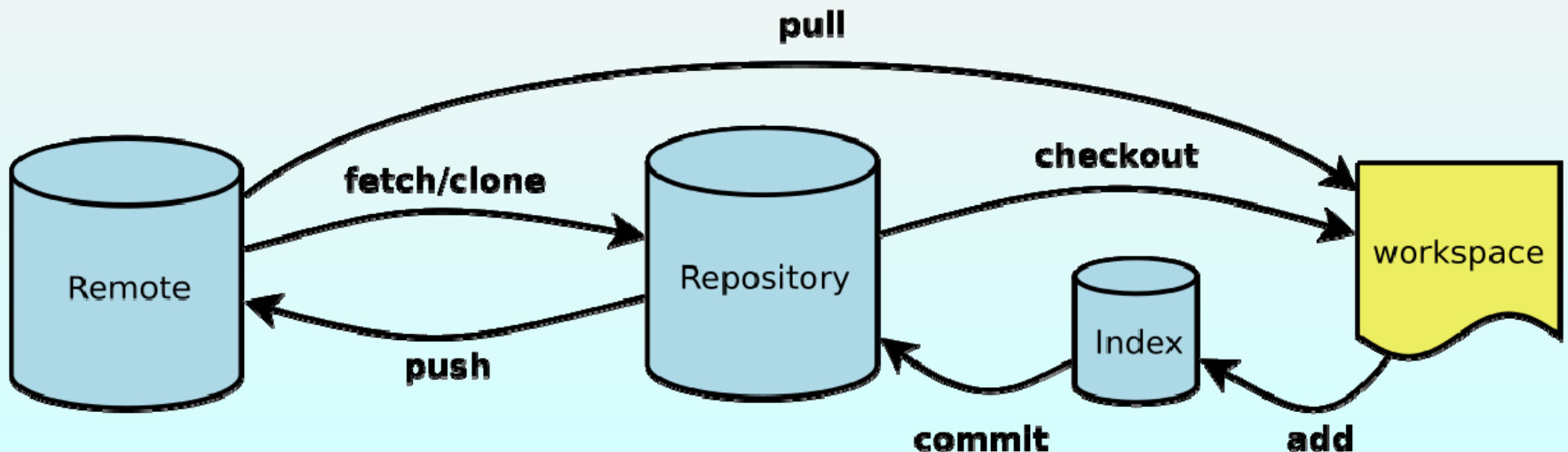
Git的基本概念

- Remote: 远程仓库
- Repository: 本地仓库
- Workspace: 工作区
- Index: 暂存区



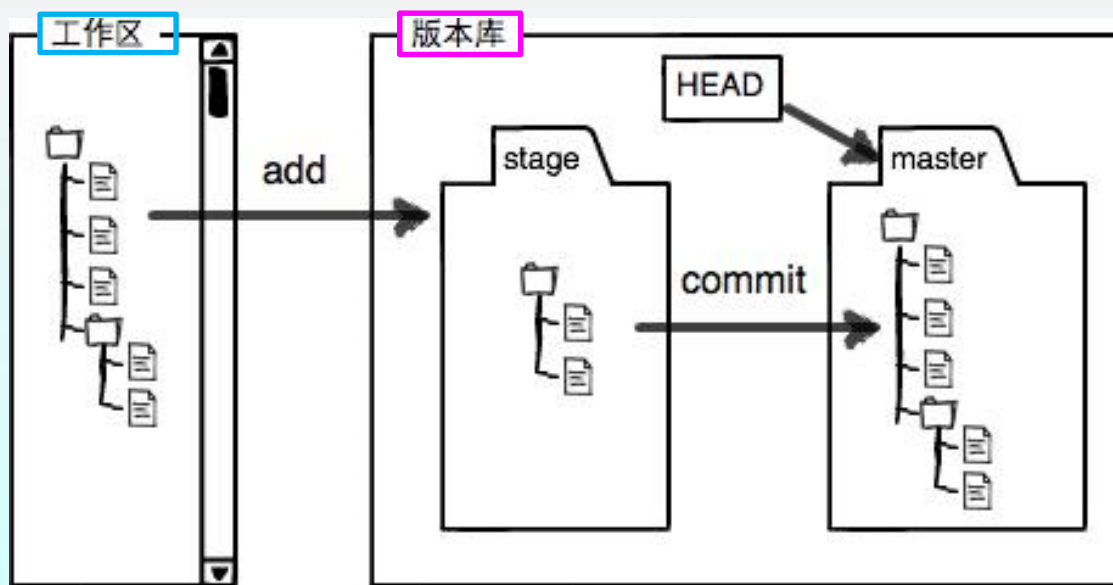
Git的重要命令

- Fetch/**clone** 更新/克隆一个远程仓库到本地
- **Push** 推送本地仓库到远程仓库
- Pull 从远程仓库更新本地文件
- Checkout 从本地仓库更新文件
- **Add** 添加本地文件到暂存区
- **Commit** 提交本地修改到本地仓库 (承诺、保证做某事)



工作区、暂存区、版本库

- **工作区** (Workspace): 当前的工作目录 (Working Directory)
- 本地仓库 (Repository): 工作区的一个隐藏目录 **.git**, 它不算工作区, 而是 Git 的版本库 (**.git** 文件夹)
- 暂存区 (index 文件): Git 的版本库里存了很多东西, 其中最重要的就是称为 **stage** 的暂存区, 还有 Git 为我们自动创建的第一个分支 **master**, 以及指向 **master** 的一个指针叫 **HEAD**



前期准备

- 安装Git, 推荐版本为: Git-2.17.0
- 在E盘新建文件夹: gitee
- 在gitee文件夹上点鼠标右键, 再, Bash Here
- Gitee文件夹就成了一个工作区



Git的执行方式

- 一般来说, 有两种执行方式
- 命令行方式(cmd)
- GUI方式 (除了需要安装Git, 还需要安装软件SourceTree, 使用方法请见最后几页PPT)
- 要求大家尽可能用命令行 (cmd)

创建本地版本库

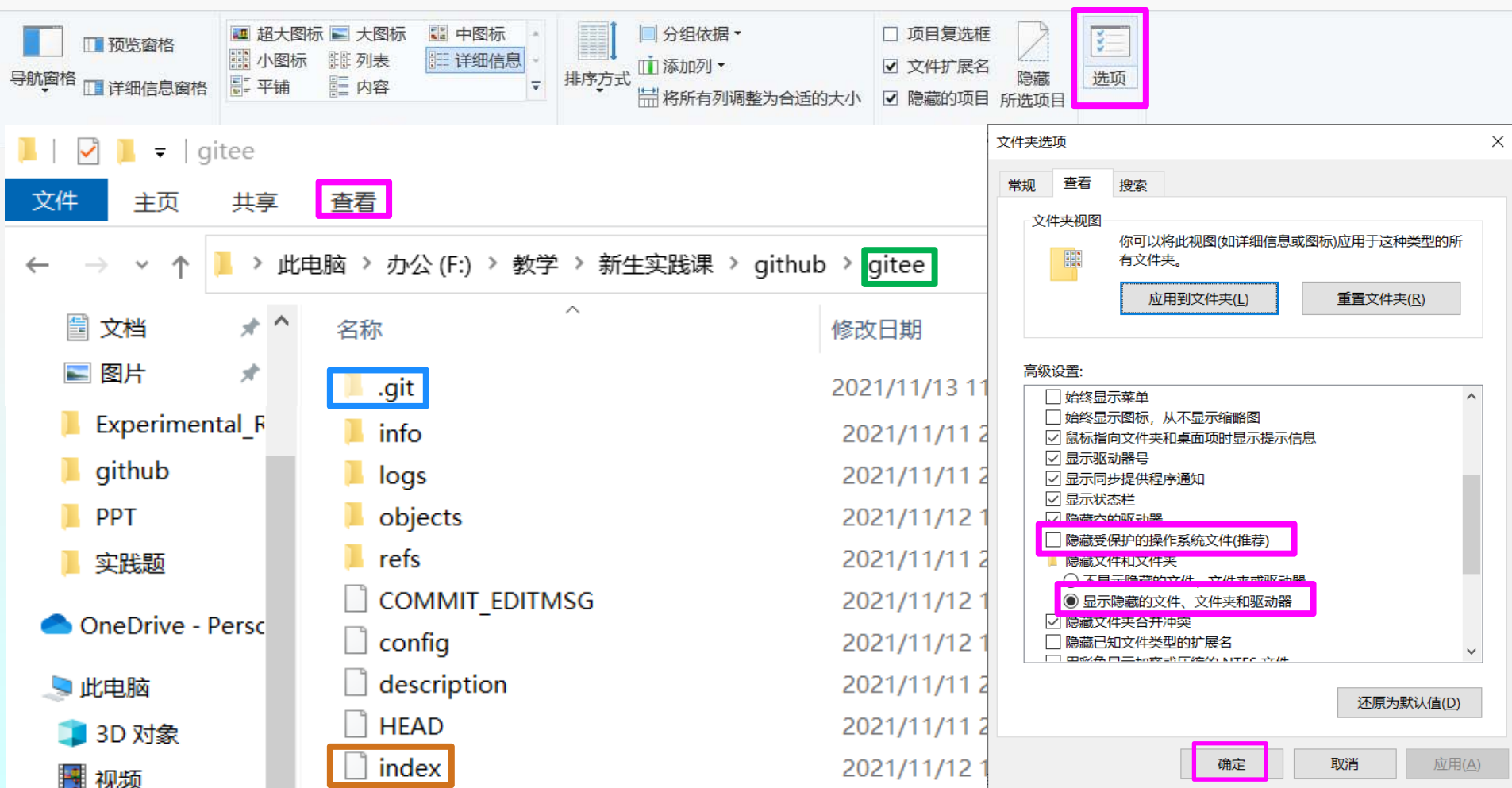
- \$ 起头, 而且是\$ git起头的命令
- 执行 `git init` 命令: 在下面的窗口中输入 `git init` 并回车!



```
MINGW64:/f/教学/新生实践课/github/gitee
jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee
$ |
```

工作区、暂存区、版本库

- 资源管理器上点查看→选项→不选隐藏受保护的操作系统文件



添加修改到暂存区域

- 创建github_menu.txt文件, 有如下方法:
 - ✓ 方法一: 在gitee文件夹下
 - ◆ 在git窗口用命令 touch github_menu.txt新建文件
 - ◆ 再用vi github_menu.txt编辑文件
 - ✓ 方法二: 新建、下载或者复制拷贝github_menu.txt
- 提交github_menu.txt到暂存区
git add github_menu.txt
- 提交工作目录中的所有文件到暂存区
git add *

文件概念及文件操作

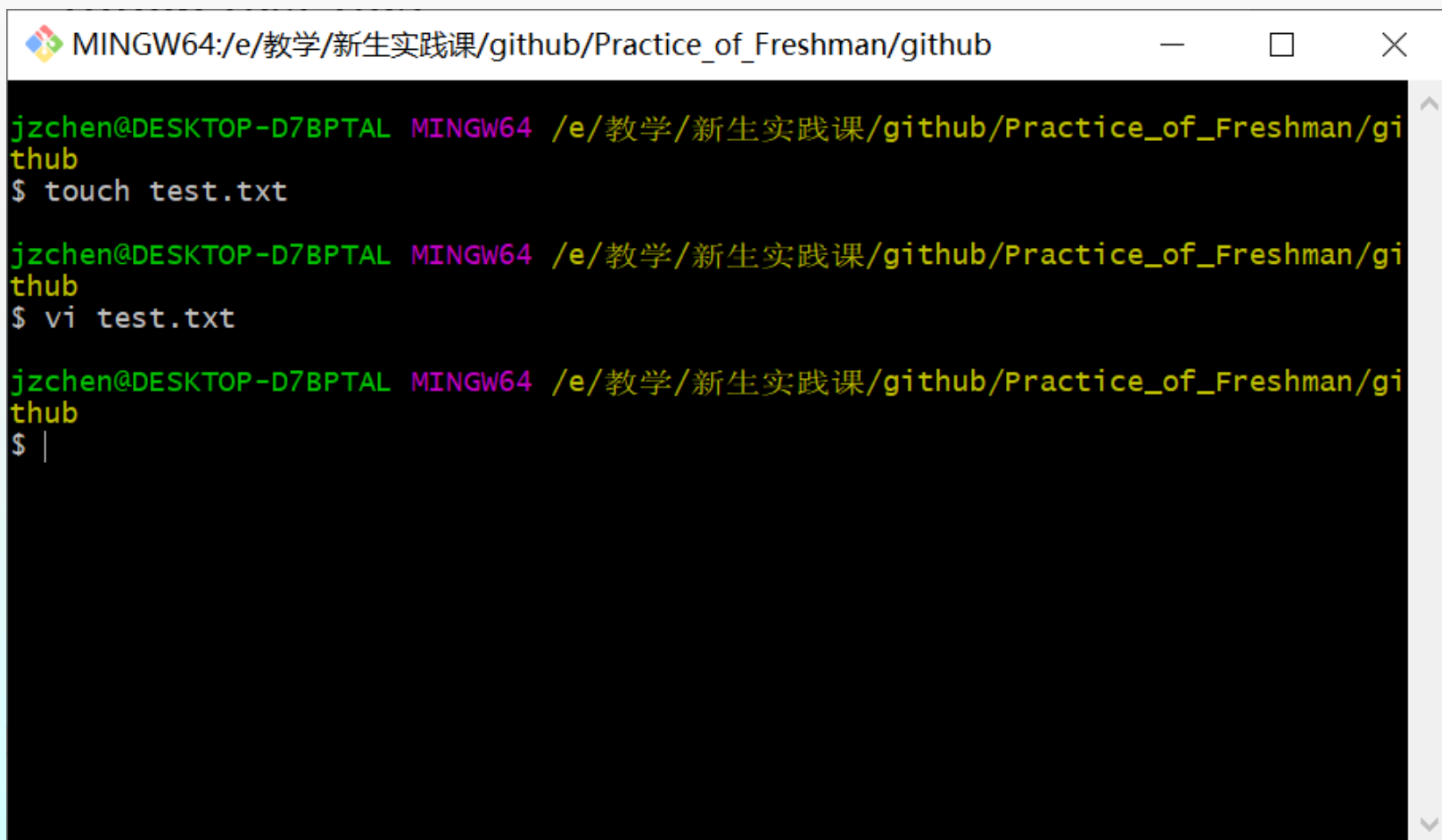
- touch test.txt



```
MINGW64:/e/教学/新生实践课/github/Practice_of_Freshman/github
jzchen@DESKTOP-D7BPTAL MINGW64 /e/教学/新生实践课/github/Practice_of_Freshman/github
$ touch test.txt
jzchen@DESKTOP-D7BPTAL MINGW64 /e/教学/新生实践课/github/Practice_of_Freshman/github
$ |
```

文件概念及文件操作

✓ 输入vi test.txt, 进入vi编辑模式



```
MINGW64:/e/教学/新生实践课/github/Practice_of_Freshman/github
jzchen@DESKTOP-D7BPTAL MINGW64 /e/教学/新生实践课/github/Practice_of_Freshman/github
$ touch test.txt

jzchen@DESKTOP-D7BPTAL MINGW64 /e/教学/新生实践课/github/Practice_of_Freshman/github
$ vi test.txt

jzchen@DESKTOP-D7BPTAL MINGW64 /e/教学/新生实践课/github/Practice_of_Freshman/github
$ |
```

文件概念及文件操作

✓ 按i键, 进入编辑模式, 输入hello

[illegible]

文件概念及文件操作

✓ 按ESC键, 退出编辑; 按英文冒号键, 再按x键, 保存退出



The screenshot shows a terminal window with the title bar "MINGW64:/e/教学/新生实践课/github/Practice_of_Freshman/github". The terminal content displays the text "hello everyone from class four to six" on the first line, followed by several lines of tilde (~) characters. The status bar at the bottom indicates the file path "<thub/Practice_of_Freshman/github/test.txt[+] [unix] (09:51 04/11/2020)1,37 全部 :x".

文件概念及文件操作

- ✓ 输入`cat test.txt`, 显示文件内容
- ✓ 输入`mkdir user`, 创建一个文件夹
- ✓ 输入`cd user`, 进入创建的文件夹`user`
- ✓ 输入`pwd`, 显示当前目录
- ✓ 输入`cp ../test.txt test1.txt`, 将上一层目录的文件`test.txt`拷贝过来并改名为`test1.txt`
- ✓ `../`表示上一层的路径

文件概念及文件操作

✓ 文件操作展示

```
MINGW64:/e/教学/新生实践课/github/Practice_of_Freshman/github/user
thub
$ cat test.txt
hello everyone from class four to six

jzchen@DESKTOP-D7BPTAL MINGW64 /e/教学/新生实践课/github/Practice_of_Freshman/github
thub
$ mkdir user

jzchen@DESKTOP-D7BPTAL MINGW64 /e/教学/新生实践课/github/Practice_of_Freshman/github
thub
$ cd user

jzchen@DESKTOP-D7BPTAL MINGW64 /e/教学/新生实践课/github/Practice_of_Freshman/github/user
$ pwd
/e/教学/新生实践课/github/Practice_of_Freshman/github/user

jzchen@DESKTOP-D7BPTAL MINGW64 /e/教学/新生实践课/github/Practice_of_Freshman/github/user
$ cp ../test.txt test1.txt

jzchen@DESKTOP-D7BPTAL MINGW64 /e/教学/新生实践课/github/Practice_of_Freshman/github/user
$
```

提交本地修改到本地仓库

- 把发生改变的文件信息提交到本地仓库
 - ✓ `git commit -m "Hello Git!"`
- 或者
 - ✓ `git commit -m "first commit"`
- 还可以查看在上次提交之后, 是否有对文件进行再次修改
 - ✓ `git status`

推送本地内容到远程仓库

- 推送本地内容时, 会将所有未推送至远程仓库的内容, 都提交到远程仓库, 它用到的命令是 **git push**, 使用方法如下: **git push** 远程仓库名 本地分支名 远程分支名, 例如:
 - ✓ **git push -u origin master**
 - ✓ 远程主机 **origin** 的 **master** 分支
 - ✓ **origin**: 远程主机
 - ✓ **master**: 分支

添加远程版本库

- 先需要有一个仓库, 有两种方法可以拥有仓库
- 但不管是哪种方法, 都需要先注册账号!!!!
 - ✓ 方法一、在gitee或者github网站上新建一个仓库
 - ✓ 方法二、先克隆别人的仓库, 比如某人的仓库url为 `https://gitee.com/chenjz70/gitee.git`, 则:
 - ◆ `git clone https://gitee.com/chenjz70/gitee`
- 然后添加远程版本库, 即执行命令: `git remote add origin https://gitee.com/chenjz70/gitee.git`

注册码云账号

- 去<https://gitee.com/> 注册账号
- 账号绝对不要有汉字和空格, 可以是zhanan22
- 需要提供邮箱, 可以用QQ信箱, 比如2276155684@qq.com, 或者lisi@hust.edu.cn
- 仓库归属如chenjz70在添加远程仓库时不能错
- 仓库名称和路径最好完全一样, 比如仓库名是MyWeb, 那么仓库的路径最好也是MyWeb, 此时, 仓库地址为<https://gitee.com/chenjz70/MyWeb.git>, 否则添加远程仓库容易出错

创建远程仓库实例

- 点符号+, 创建仓库

The screenshot shows the Gitee user profile for 'chenjz70'. The top navigation bar includes the Gitee logo, links for '开源软件', '企业版', '高校版', '私有云', '博客', 'Go', and '我的'. A search bar is on the right. The user's profile section on the left shows a blue circular avatar with a white 'C', the username 'chenjz70', and social statistics: 2 Stars, 7 Watches, 69 Followers, and 1 Following. The main content area is titled '热门项目' (Popular Projects) and features a grid of project cards. A pink box highlights the '+' icon in the top right corner of the navigation bar, which is used to create a new repository. The project cards include 'Practice_of_Freshman', 'Gaze Estimation', 'experimental-report', 'HUSTGraduateThesis', 'mygitee', and 'gitee'.

Navigation Bar: gitee 开源软件 企业版 高校版 私有云 博客 Go 我的 搜开源

User Profile: chenjz70 @chenjz70 chenjz70 暂无简介 个人设置

Statistics: 2 Stars 7 Watches 69 Followers 1 Following

Project Grid:

- Practice_of_Freshman**
Practice_of_Freshman
4 75 0
- Gaze Estimation**
Gaze Estimation via Bilinear Pooling-Based Attention Networks. Dakai Ren, Jiazhong Chen, Jian Zhong, Zhaoming Lu, Tao Jia, an...
1 4 73
- experimental-report**
大学生实践课实验报告latex模板, 没有什么非法侵权内容, 模板是作者自己开发的
5 5 21
- HUSTGraduateThesis**
hust本科学位论文latex模板
</> TeX/LaTeX
1 2 2
- mygitee**
1 2 1
- gitee**
py的几个例子供大家批评
1 1 0

创建远程仓库实例

- 填好仓库名与路径(名称要一致), 点创建

新建仓库

在其他网站已经有仓库了吗? [点击导入](#)

仓库名称 * ✓

MyWeb

归属

chenjz70 / 路径 * ✓

MyWeb

仓库地址: <https://gitee.com/chenjz70/MyWeb>

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

创建远程仓库实例

● 创建后的页面会给出一些常用命令

快速设置— 如果你知道该怎么操作, 直接使用下面的地址

HTTPS SSH `https://gitee.com/chenjz70/MyWeb.git` 

我们强烈建议所有的git仓库都有一个 `README`, `LICENSE`, `.gitignore` 文件

初始化 `readme` 文件

Git入门? [查看](#) [帮助](#), [Visual Studio](#) / [TortoiseGit](#) / [Eclipse](#) / [Xcode](#) 下如何连接本站, [如何导入仓库](#)

简易的命令行入门教程:

Git 全局设置:

```
git config --global user.name "chenjz70"  
git config --global user.email "chenjz70@163.com"
```

创建 git 仓库:


```
mkdir MyWeb  
cd MyWeb  
git init  
touch README.md  
git add README.md  
git commit -m "first commit"  
git remote add origin https://gitee.com/chenjz70/MyWeb.git  
git push -u origin "master"
```

已有仓库?

```
cd existing_git_repo  
git remote add origin https://gitee.com/chenjz70/MyWeb.git  
git push -u origin "master"
```

创建远程仓库实例

- 仓库是私有的, 归属中看不见, 要点仓库才能看见



chenjz70
@chenjz70
chenjz70 暂无简介

个人设置

2 Stars 7 Watches 69 Followers 1 Following

概览 仓库 7 星选集

热门项目

Practice_of_Freshman
Practice_of_Freshman

4 75 0

Gaze Estimation
Gaze Estimation via Bilinear Pooling-Based Attention Networks.
Dakai Ren, Jiazhong Chen, Jian Zhong, Zhaoming Lu, Tao Jia, an...

1 4 73

experimental-report
大学生实践课实验报告latex模板, 没有什么非法侵权内容, 模板是作者自己开发的

5 5 21

HUSTGraduateThesis
hust本科学位论文latex模板

</> TeX/LaTeX
1 2 2

mygitee

1 2 1

gitee
py的几个例子供大家批评

1 1 0

自定义精选项目

创建远程仓库实例

- 点管理, 可以设置为开源的

chenjz70 / MyWeb

Watching 1

Star 0

Fork 0

代码

Issues 0

Pull Requests 0

Wiki

统计

流水线

服务

管理

master 分支 1 标签 0

+ Pull Request

+ Issue

文件

Web IDE

克隆/下载

chenjz70	1st commit	c0485e1	7分钟前	1 次提交
image	1st commit		7分钟前	
web	1st commit		7分钟前	
cuikun.txt	1st commit		7分钟前	
index.html	1st commit		7分钟前	
lvzhipeng.txt	1st commit		7分钟前	

添加一个 README.md 文件, 帮助感兴趣的人了解。 [添加 README](#)

简介

暂无描述

暂无标签

JavaScript 等 2 种语言

发行版

暂无发行版, [创建](#)

贡献者 (1)

C

近期动态

C

5分钟前推送了新的 master 分支

C

26分钟前创建了仓库

Clone远程版本库

- 将远程版本库clone到本地
 - ✓ `git clone https://gitee.com/chenjz70/experimental-report`



Starred
47



Fork
159




捐赠
0 人次


 **chenjz70 / SeDemo** 

forked from 华中科技大学-范晔斌 / SeDemo

 代码

 Issues 0




 Pull Requests 0

 Wiki

推送步骤拆解


- 建一个名字为gitee的仓库



https://gitee.com/projects/new


G gitee 开源软件 企业版 特选 高校版 私有云 博客 我的   

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 * 


 chenjz70 /

仓库地址: https://gitee.com/chenjz70/gitee

仓库介绍 0/100

☐ 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) 

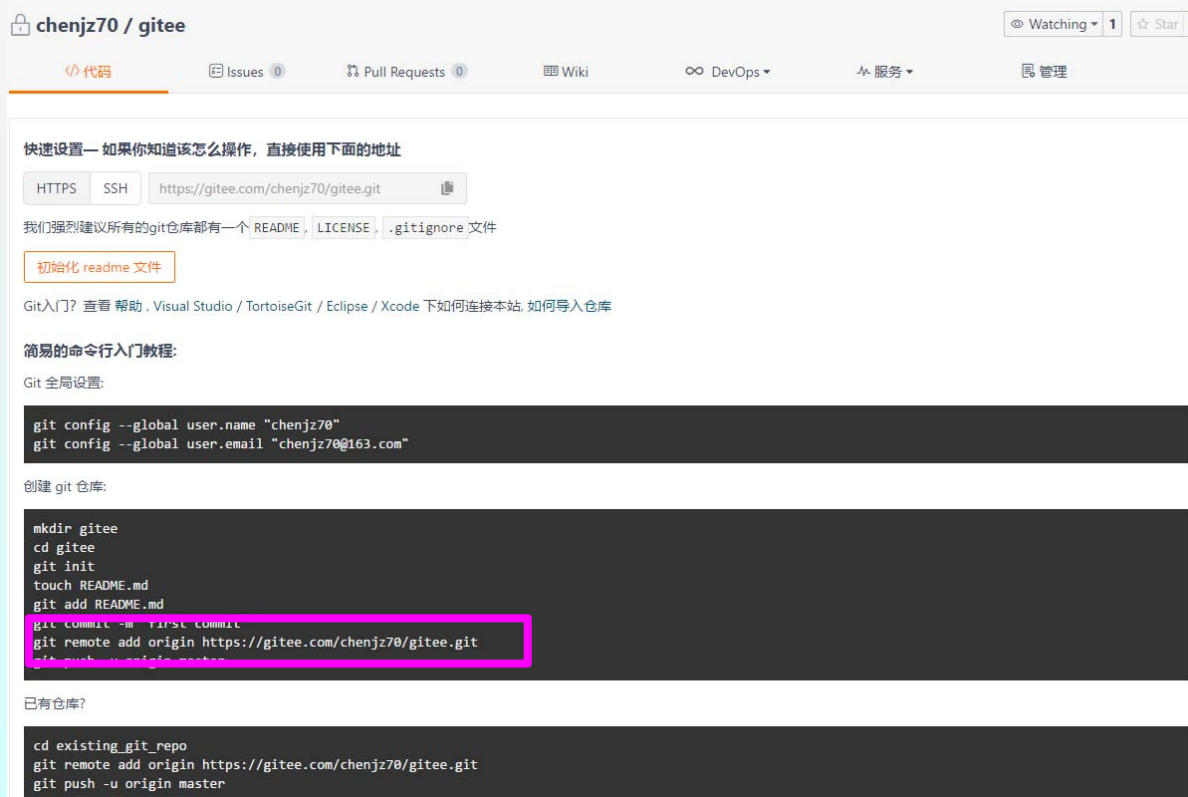
☐ 初始化仓库 (设置语言、.gitignore、开源许可证)

☐ 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)

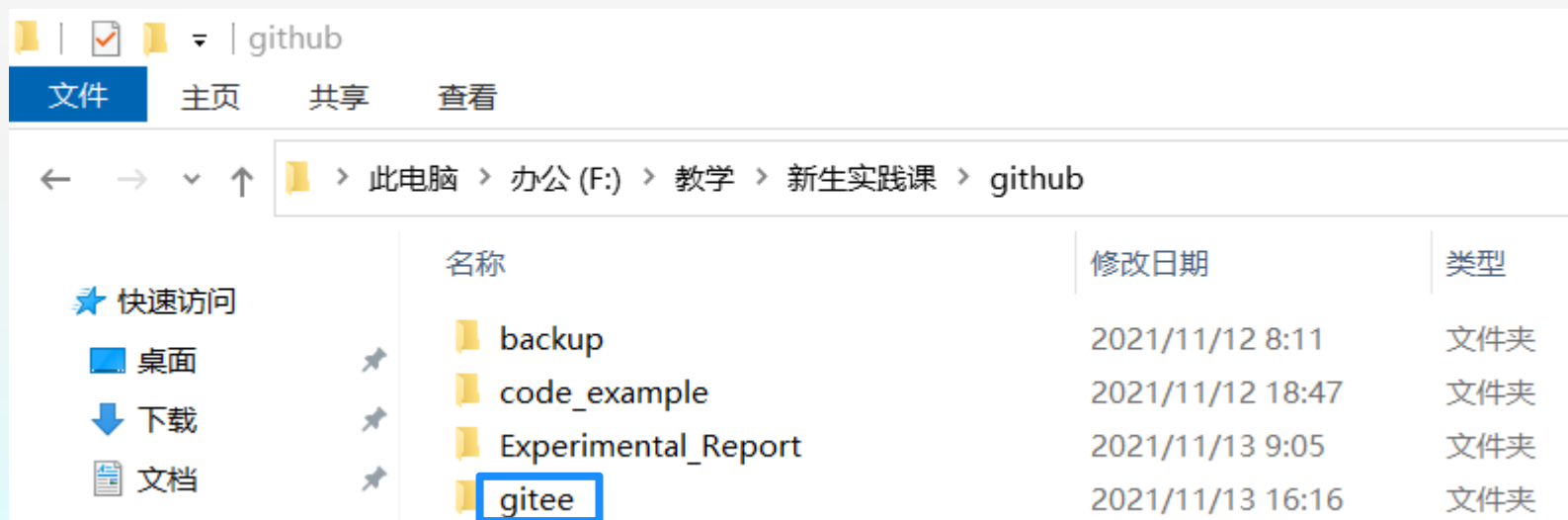
推送步骤拆解

- 请记住创建成功后页面上的仓库信息, 比如
`git remote add`
`origin https://gitee.com/chenjz70/gitee.git`



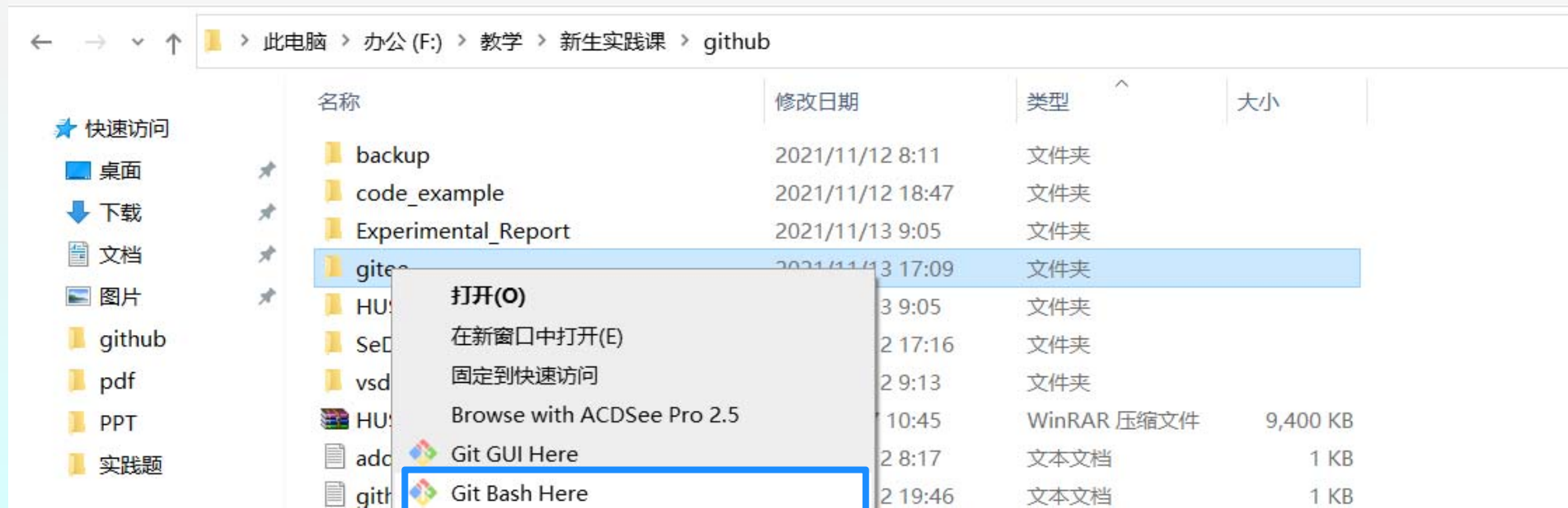
推送步骤拆解

- 在E盘建一个gitee文件夹
- 随便拷贝几个.py文件进gitee文件夹



推送步骤拆解

- 在gitee文件夹上点鼠标右键→点git bash here
- 严禁在工作区内新建工作区



推送步骤拆解

- 进入Git命令行界面
- 输入git init命令并回车



A screenshot of a Windows command prompt window. The title bar at the top reads "MINGW64:/f/教学/新生实践课/github/gitee" and includes standard window controls (minimize, maximize, close). The command prompt shows the user "jzchen@DESKTOP-4SUJG03" in a "MINGW64" environment at the path "/f/教学/新生实践课/github/gitee". The command "\$ git init" has been entered and is ready to be executed. The background of the command prompt is black, and the text is white.

```
MINGW64:/f/教学/新生实践课/github/gitee
jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee
$ git init
```

推送步骤拆解

- `git config --global user.name "你的账号"`并回车
- `git config --global user.email "你的邮箱"`并回车

A screenshot of a Windows terminal window titled "MINGW64:/f/教学/新生实践课/github/gitee". The terminal shows the following commands and output:

```
jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee
$ git init
Initialized empty Git repository in F:/教学/新生实践课/github/gitee/.git/


jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee (master)
$ git config --global user.name "chenjz70"

jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee (master)
$ git config --global user.email "chenjz70@163.com"

jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee (master)
$
```

推送步骤拆解

- `git add *` 并回车
- `git commit -m "first commit"` 并回车



```
MINGW64:/f/教学/新生实践课/github/gitee

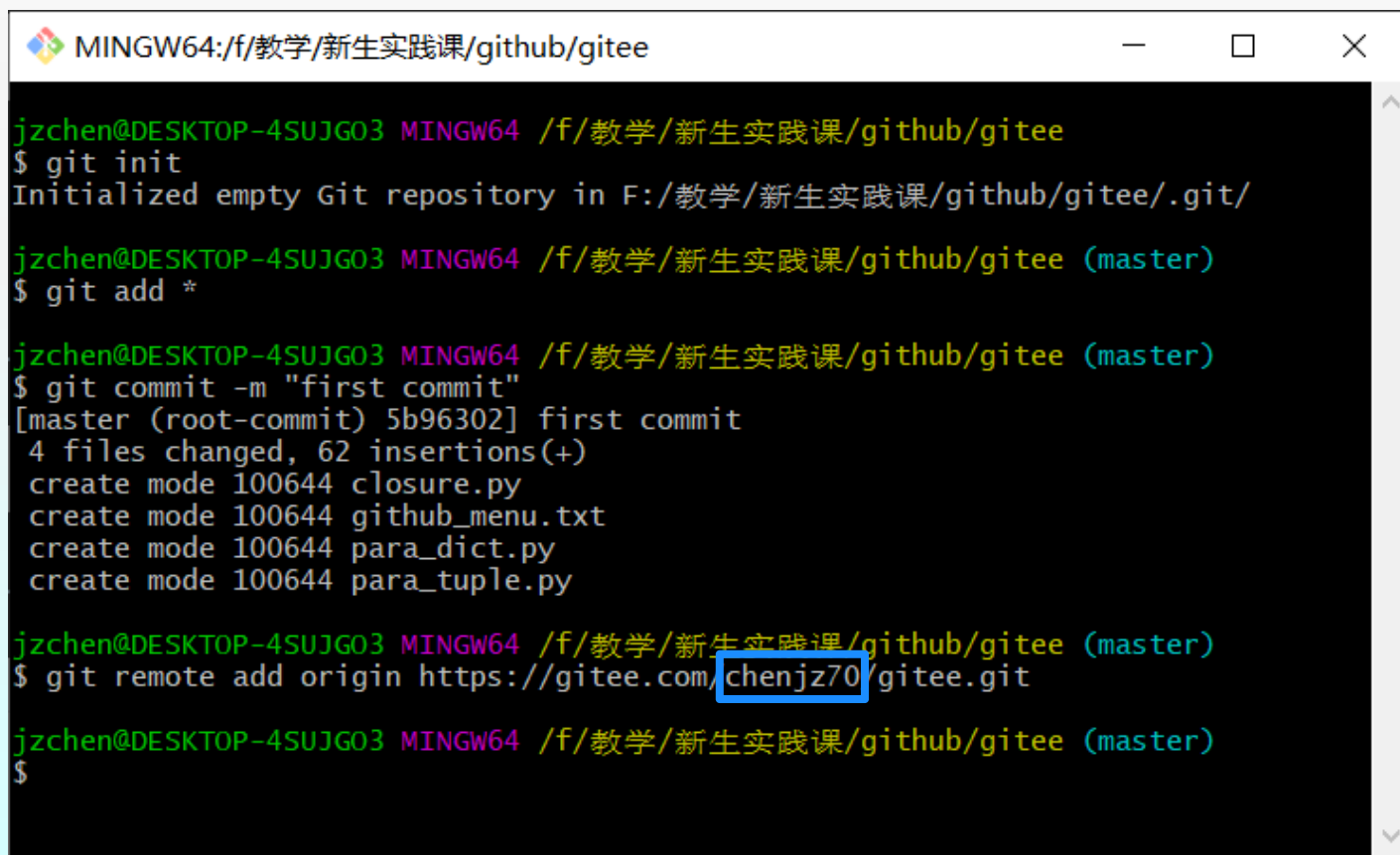
jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee (master)
$ git add *

jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee (master)
$ git commit -m "first commit"
[master (root-commit) 45f2e54] first commit
1 file changed, 20 insertions(+)
create mode 100644 github_menu.txt

jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee (master)
$
```

推送步骤拆解

- `git remote add origin https://gitee.com/你的账号(不是昵称是账号)/gitee.git`, 并回车



```
MINGW64:/f/教学/新生实践课/github/gitee
jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee
$ git init
Initialized empty Git repository in F:/教学/新生实践课/github/gitee/.git/

jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee (master)
$ git add *

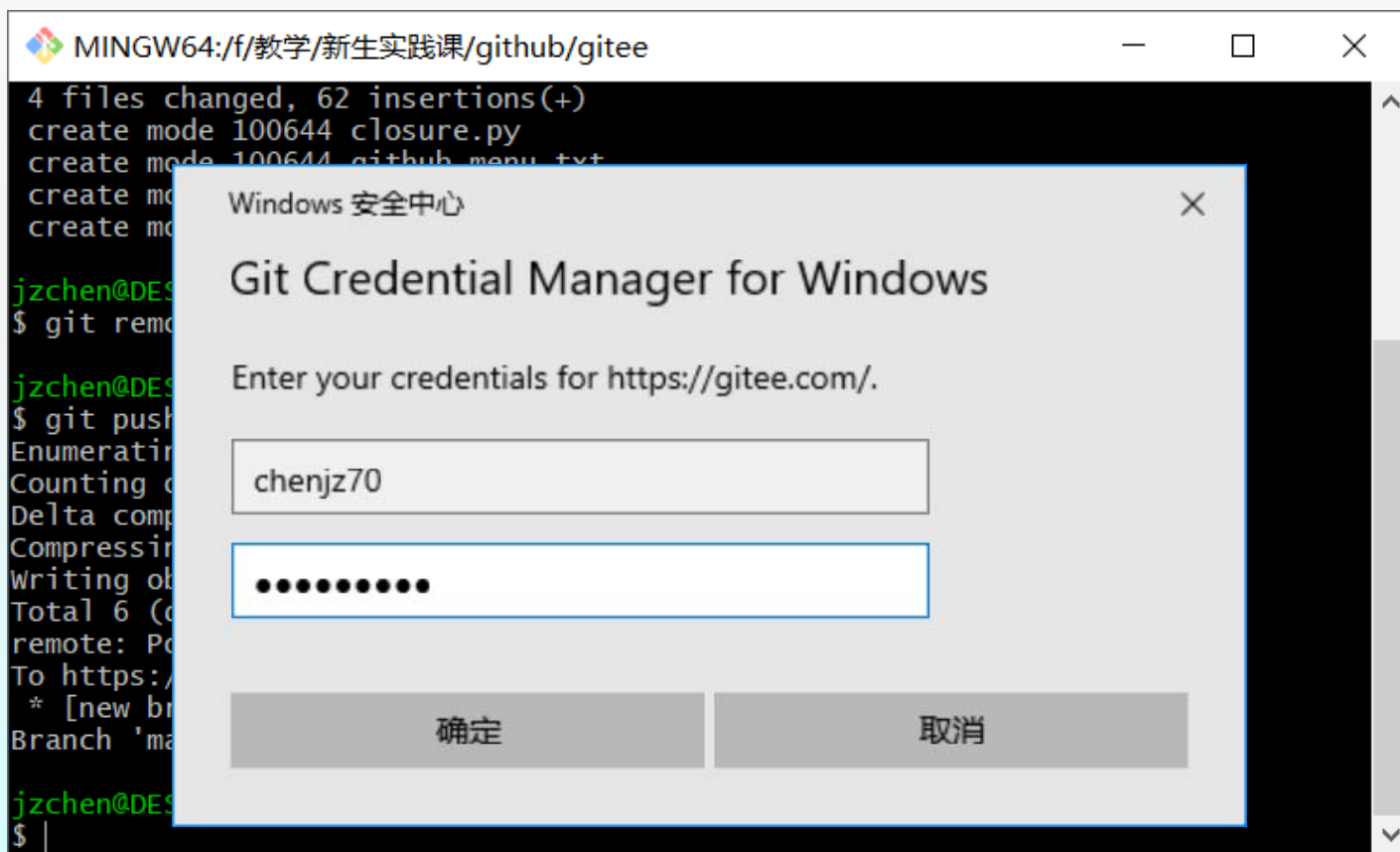
jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee (master)
$ git commit -m "first commit"
[master (root-commit) 5b96302] first commit
4 files changed, 62 insertions(+)
create mode 100644 closure.py
create mode 100644 github_menu.txt
create mode 100644 para_dict.py
create mode 100644 para_tuple.py

jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee (master)
$ git remote add origin https://gitee.com/chenjz70/gitee.git

jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee (master)
$
```

推送步骤拆解

- `git push -u origin master`, 并回车, 输入账号密码



推送步骤拆解

- `git push -u origin master`, 输入账号密码后点确定

```
MINGW64:/f/教学/新生实践课/github/gitee
4 files changed, 62 insertions(+)
create mode 100644 closure.py
create mode 100644 github_menu.txt
create mode 100644 para_dict.py
create mode 100644 para_tuple.py


jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee (master)
$ git remote add origin https://gitee.com/chenjz70/gitee.git

jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee (master)
$ git push -u origin master
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 6 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 1.36 KiB | 463.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
remote: Powered by GITEE.COM [GNK-6.2]
To https://gitee.com/chenjz70/gitee.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.

jzchen@DESKTOP-4SUJG03 MINGW64 /f/教学/新生实践课/github/gitee (master)
$ |
```

推送步骤拆解

- 在gitee网站上查看是否推送成功

 **gitee** 开源软件 企业版 ^{特惠} 高校版 私有云 博客 我的 ▾

搜开源

🔔 0 🏠 + C ▾


chenjz70 / **gitee**

👁 Watching ▾ 1 ☆ Star 0 🍴 Fork 0

[📄 代码](#) [🗨 Issues 0](#) [🔗 Pull Requests 0](#) [📖 Wiki](#) [📊 统计](#) [∞ DevOps ▾](#) [🔧 服务 ▾](#) [👤 管理](#)

master ▾ 🔗 分支 1 🔖 标签 0

[+ Pull Request](#) [+ Issue](#) [文件 ▾](#) [Web IDE](#) [克隆/下载 ▾](#)

 chenjz70 first commit 5b96302 2分钟前

👁 1 次提交

📄 closure.py	first commit	2分钟前
📄 github_menu.txt	first commit	2分钟前
📄 para_dict.py	first commit	2分钟前
📄 para_tuple.py	first commit	2分钟前

添加一个 README.md 文件，帮助感兴趣的人了解。 [添加 README](#)

简介

暂无描述


暂无标签

📄 Python

发行版

暂无发行版， [创建](#)

贡献者 (1)

 C

🔗 全部

强制推送

- `git push -force`
- 虽然强制推送命令在某些情况下非常有用，但它也存在一定的风险：
 - ✓ 强制推送可能会导致远程代码库中的修改丢失，因此在使用之前需要慎重考虑
 - ✓ 如果多人同时使用强制推送命令，可能会导致代码库的混乱和冲突，因此需要与团队成员保持良好的沟通和协调

强制推送的替代方案

- `git pull -rebase origin master`
- 在进行强制push之前, 先使用git pull命令, 将远程仓库中的代码拉取到本地, 解决代码冲突后再进行强制push操作

输错账号密码解决

- 如果不小心输错账号密码
- 或者机器上已经有了别人的账号与密码
- **win+r→control→用户账号→管理windows凭据→编辑修改gitee账号或密码**

版本回退步骤拆解

- 在已有的本地仓库中**修改**某个txt或者py文件
- 该文件**还没有**push到远程仓库
- 假如该文件叫Readme.txt
- 此时想恢复修改前的版本, 怎么办?
 - ✓ `git add Readme.txt`
 - ✓ `git reset --hard`
- 这样就在忘记修改了什么地方时, **Git**就可以帮咱来回忆!!

欢迎大家star老师的仓库

- **Watch:** 对于别人的项目,默认自己都处于 **Not watching** 的状态,当选择 **Watching**,表示以后会关注这个项目的所有动态,这个项目以后只要发生变动,如被别人提交了 **pull request**、被别人发起了**issue**等等情况,都会在自己的个人通知中心,收到一条通知消息,如果设置了个人邮箱,那么邮箱也可能收到相应的邮件
- **Star:** 解释为“关注”或者“点赞”更合适,当点击 **star**,表示喜欢这个项目或者理解成朋友圈的点赞,表示对这个项目的支持
- **Fork:** (复刻,又译作派生、分支)当选择 **fork**,相当于拥有一份原项目的拷贝,当然这个拷贝只是**针对当时的**项目文件,如果后续原项目文件发生改变,必须通过其他方式去同步

拉取远程分支到本地

- 拉取远程仓库的内容到本地, 需要使用git pull命令, 其命令格式为: `git pull 远程主机名 远程分支名 本地分支名`, 例如:
 - ✓ `git pull origin`
 - ✓ 将远程主机 origin 的 master 分支拉取过来, 与本地的 master 分支合并: `git pull origin master :master`
 - ◆ 如果远程分支是与当前分支合并, 则冒号后面的部分可以省略

创建本地分支

- 使用分支意味着可以把工作从开发主线上分离开, 以免影响开发主线
- 当需要创建一个新的本地分支的时候, 可以使用 `git branch` 命令, 其具体使用格式为: `git branch` 新的分支名字, 例如:
 - ✓ `git branch gitTraining`
- 创建新分支的同时切换到这个新的分支, 有一个更为简洁的命令: `git checkout -b`, 它的使用格式为: `git checkout -b` 新的分支名字, 例如:
 - ✓ `git checkout -b gitTraining`

删除本地分支

- 删除本地分支, 需要用到git branch命令, 且需要-D参数, 具体命令格式为: git branch -D 需要删除的分支的名字, 例如删除分支develop:
✓ git branch -D develop

删除远程分支

- 删除分支用到的git命令是git push, 有两种方法:
 - ✓ 1.通过推送空分支到远程分支, 实现删除, 推送空分支实现删除的方法是: git push 远程主机名 :远程分支, 例如:
 - ◆ git push origin :develop
 - ✓ 2.通过delete参数删除远程分支, 命令格式为: git push 远程主机名 --delete 远程分支名, 例如:
 - ◆ git push origin --delete develop

本地分支合并

- 分支合并需要用到git merge命令, 具体的命令格式为: git merge 需要合并的分支, 例如, 将mygit分支合并到当前主分支master:
 - ✓ git merge mygit

回到前一次提交

- 版本回退可以用git revert命令, 例如:
 - ✓ `git revert HEAD` // 撤销前一次 commit
 - ✓ `git revert HEAD^` // 撤销前前一次 commit
 - ✓ 按英文冒号键, 再按x键, 保存退出
- 版本回退还可以用git reset命令, 例如:
 - ✓ `git reset HEAD` // 撤销前一次 commit
- git revert是用一次新的commit来回滚之前的commit, git reset是直接删除指定的commit

三种不同的撤销修改

- 如果只是工作区有了修改, 可以使用 `git checkout` 进行撤销, 例如:
 - ✓ `git checkout -- hello`, 通过这种方式, 就可将 `hello` 文件自上个 `commit` 之后, 尚未 `add` 进暂存区的修改丢弃
- 当将有错误的文件 `add` 进暂存区后, 可以使用 `git reset` 丢弃修改, 例如:
 - ✓ `git reset HEAD 文件名`, 但此时修改仍旧保留在工作区
- 如果尚未 `add` 进暂存区, 则可以使用:
 - ✓ `git reset --hard HEAD` 这样就能彻底丢弃修改, 即将修改从暂存区及工作区彻底删除

删除文件

- 从仓库中删除文件的一般过程为：

1.git rm <--cached> 文件名

2.git commit -m "提交信息"

- 例如：

git rm --cached helloGit

git commit -m "删除helloGit"

创建标签

- 创建标签的命令格式为: `git tag 标签名 commitID`, 例如:
 - ✓ `git tag v1.0`
 - ✓ Git 就是对某次commit的一个标识, 相当于起了一个别名
 - ✓ 例如, 在发布某个版本的时候, 针对最后一次commit起一个v1.0这样的标线来标识

推送指定标签

- 推送指定标签到远程仓库的Git命令如下: `git push 远程主机名 tag名`, 其中远程主机名为远程Git版本库对应的主机名, tag名为准备推送的标签名, 例如:

✓ `git push origin v1.0`

推送全部标签

- 推送全部标签需要用到: `git push 远程主机名 -tags`, 例如:
 - ✓ `git push origin --tags`

删除标签

- 删除本地标签, 需要用到的命令格式为: `git tag -d` 标签名, 例如:
 - ✓ `git tag -d v1.0`
- 删除远程标签, 可以使用: `git push origin --delete` tag标签名, 或者使用:
 - ✓ `git push origin :refs/tags/<tagname>`
- 即推送一个空 tag名到远程仓库, 其中<tagname>指某个标签的名字
- 在Git v1.7.0 之后, 可以使用如下语法删除远程标签:
 - ✓ `git push origin --delete tag` 标签名

冲突处理

- 冲突包括以下两种：内容冲突与树冲突
- 内容冲突：针对版本库中某个文件的某项内容，不同的操作对其做了不同的修改，以致于在合并不同的操作时发生矛盾
- 解决方法：
 - ✓ 1. 手动编辑冲突区域
 - ✓ 2. 执行 `git add`, 将编辑提交到暂存区
 - ✓ 3. 执行 `git commit`, 将编辑提交到本地仓库以解决冲突
- 树冲突：文件名修改造成的冲突，称为树冲突
- 解决方法：删除冲突文件，添加正确文件并提交

强制操作

- 强制推送: 如果远程分支的内容需要被覆盖, 这个时候需要进行强制推送, 使用本地内容去覆盖该分支, 例如:
 - ✓ `git push origin <your_branch_name> -f`
- 强制合并: 如果本地分支的内容需要被远程内容覆盖, 这个时候需要强制合并远程分支内容到本地
- 强制删除: 如果需要强制删除版本库、暂存区或者工作区的内容时, 就需要强制删除
 - ✓ 比如之前介绍的`checkout`, 就可以使用`-f`参数, 强制丢弃本地修改

忽略文件

- 如何忽略文件: 在Git工作区的根目录下, 创建一个特殊的.gitignore文件, 把要忽略的文件名或者文件名的通配符填进去, 然后将.gitignore提交到本地仓库, 这样Git就会在用户添加或者提交时, 自动忽略这些文件
- 自定义忽略文件: 如果需要自己定义忽略哪些文件, 就需要将其添加到.gitignore文件中
- 可以使用文件的全称, 或者使用正则匹配的通配符

合并远程分支

- 合并远程分支的一般步骤为：
 - ✓ 第一步, 分别获取远程分支内容到本地
 - ✓ 第二步, 在本地将两个分支合并
 - ✓ 第三步, 将合并后的本地分支推送到远程分支, 完成合并

rebase的基本操作

- rebase的基本操作是将某个分支的修改到指定分支, 其命令格式为: `git rebase 基分支 源分支`
- 其中“基分支”是我们的新的“基”, 而“源分支”就是需要进行变基操作的分支, 这样就能实现将源分支变基到基分支, 例如:
 - ✓ `git rebase master develop`

rebase的基本操作

- **git rebase master**
- 以上语句就能实现将develop变基到master分支
- 如果是将当前分支变基到指定分支, 则可以直接使用: **git rebase 基分支**, 这一命令**默认**将当前分支变基到“**基分支**”, 如果当前处于develop分支, 则其使用示例如下:
 - ✓ **git rebase master**

储藏之保存及恢复

- 储藏可以获取工作目录的中间状态(包括修改过的被追踪的文件和已经暂存的变更),并将其保存到一个未完结变更的堆栈中,而且随时可以重新应用
- 当不想提交,也不想丢弃当前工作区中的内容,而想切换到其他分支的时候,可以使用储藏命令先暂存工作区中的内容
- 然后,再回到当前分支的时候,将储藏起来的内容,恢复到工作区之后,即可恢复之前的工作

储藏之保存及恢复

- 储藏的保存用到的命令是`git stash`, 只需在当前分支执行此命令, 即可将当前工作区的内容保存起来
- 当需要再次应用被保存的内容的时候, 可执行`git stash apply`
- Stash: 存放; 贮藏; 隐藏

GUI方式使用Git

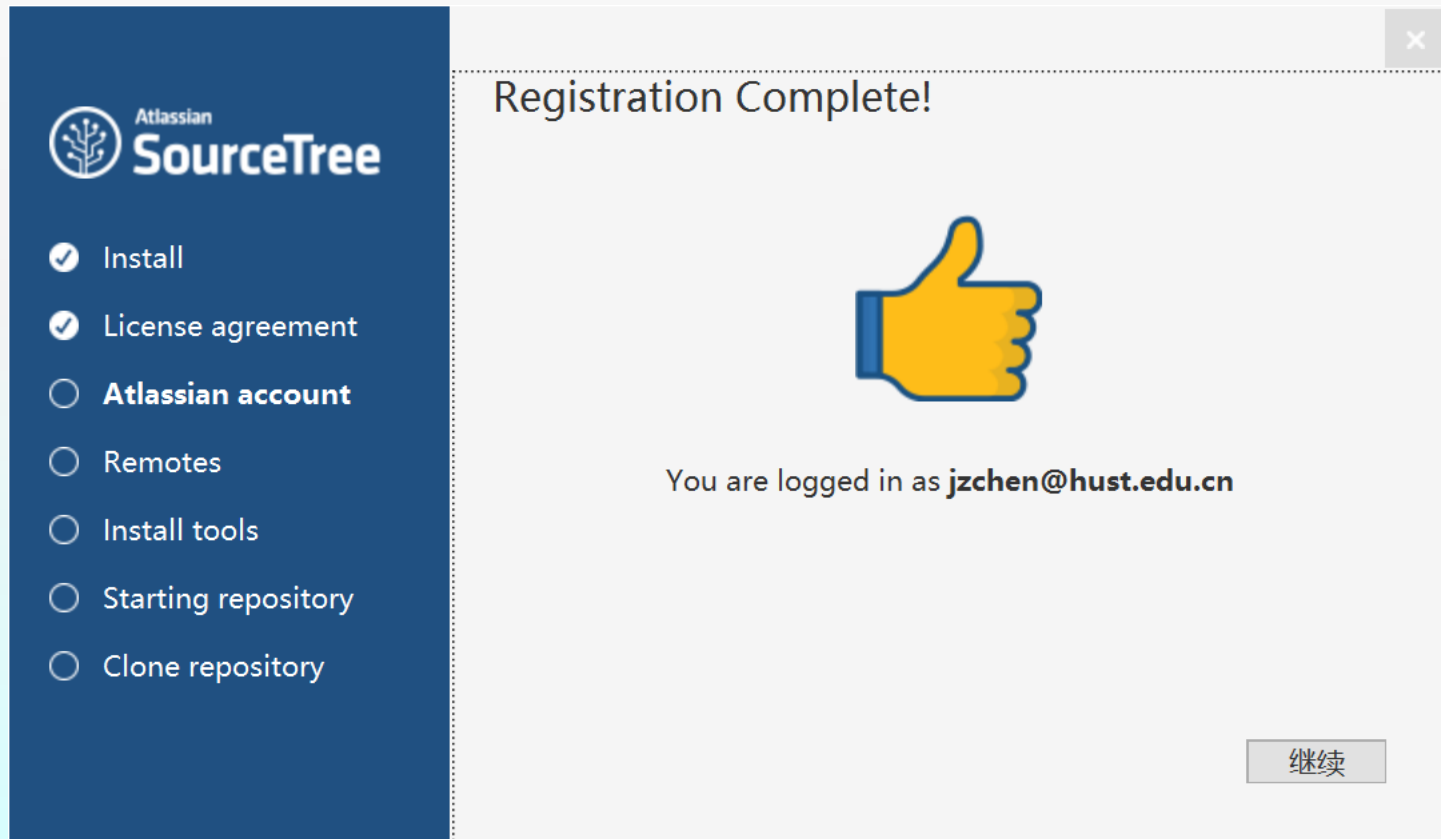
- 主流IDE中均有Git的集成
- 集成到文件管理器中的TortoiseGit
- 跨平台的SoureTree
 - ✓ 安装SourceTree (向导中选择安装内嵌的Git)。
 - ✓ 基本操作:
 - ◆ 建立本地目录 (例如D:\Work)。
 - ◆ 将码云上的远程仓库克隆到本地目录 (Clone操作, 需要用到码云注册时的账号密码)。
 - ◆ 本地文件的添加或修改 (Add、Commit操作)。
 - ◆ 推送到远程仓库 (Push操作)。

SourceTree

- SourceTree 是 Windows 和 OS X 下免费的 Git 和 Hg 客户端
- 支持创建、克隆、提交、push、pull 和合并等操作
- 从远程库克隆Clone: Clone就是将远程库的代码拷贝到本地

SourceTree

- 跨平台的SourceTree

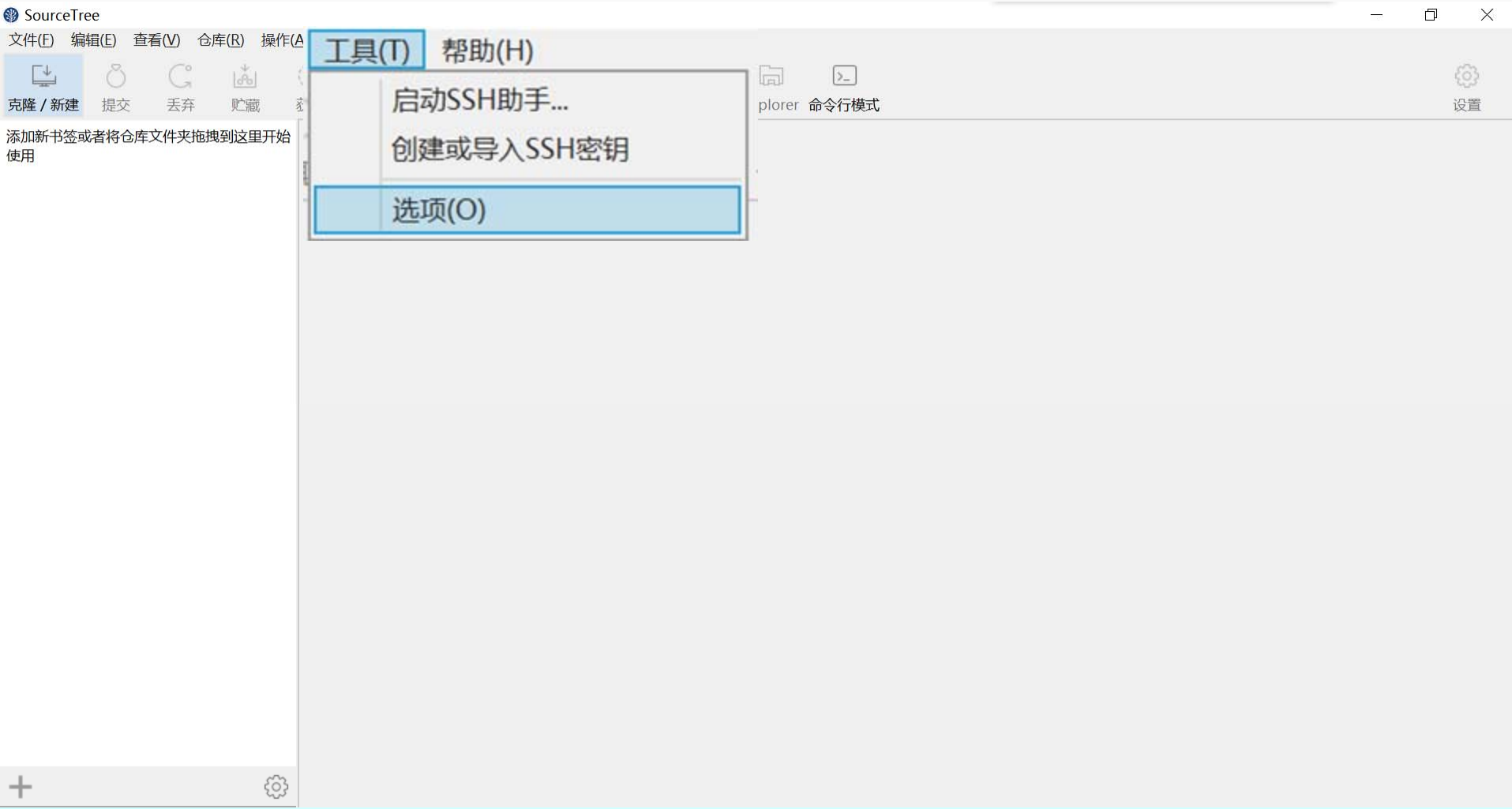


SourceTree

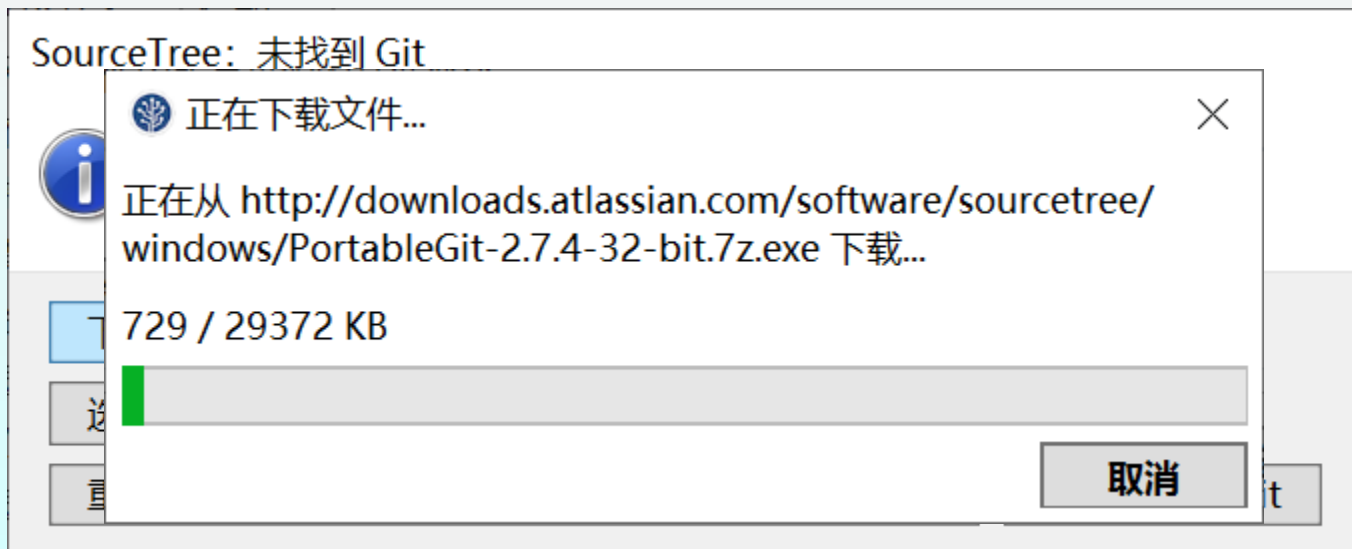
- 跨平台的SourceTree



SourceTree



SourceTree



SourceTree

选择

一般

全局忽略

推送

☐ 对跟踪分支进行推送

☒ 在推送前进行合并

☒ 推送前进行提交

☐ 禁用快速推送

☐ 合并时创建新分支

☐ 显示未跟踪文件

☒ 使用默认用户名

☐ 显示未跟踪文件

Git 版本

内嵌

使用

一般

比较

Git

Mercurial

自定义操作

验证

网络

SourceTree会在你登录远程服务器的时候记住你的用户名和密码，你可以在这里对相关记录进行手动管理。

已存密码

用户名	主机	密码
chenjz70	163.com@github.com	xxxxxx
chenjz70	github.com	xxxxxx

编辑

删除

默认用户名

对于一个给定的主机，如果在URL中未指定用户名，SourceTree将使用下列默认用户名，以免每次都向您索要。

主机	用户名
----	-----

编辑

删除

确定

确定

SourceTree

- 缺点: 不能用git把文件推到github. 独立安装

克隆 / 添加 / 创建仓库

 克隆仓库

 添加工作副本

 创建新仓库

源路径 / URL: ... 

仓库类型:  这是一个 Git 仓库

目标路径: ...

高级选项

书签

☒ 将此仓库存入书签

名字:

文件夹:

克隆

取消

进一步了解Git

- <https://www.liaoxuefeng.com/wiki/896043488029600>