

# 为什么要有代码管理

---

因为有小明、笨笨、志强

因为开发过程中有很多人分工合作 大家都要修改代码 而且代码修改后 可能还要还原回去 不一定每一次修改都是正确的,再加上不同时间要发布不同的版本 所以代码必须进行管理要不全乱了。

代码管理的好处

- 多人协作
- 历史版本回溯
- 多份备份
- 进度配合和管理
- 程序员必备技能

## git诞生

---

1991年Linux诞生，从此，Linux系统不断发展，已经成为最大的服务器系统软件了。

Linus虽然创建了Linux，但Linux的壮大是靠全世界热心的志愿者参与的，这么多人在世界各地为Linux编写代码，那Linux的代码是如何管理的呢？

事实是，在2002年以前，世界各地的志愿者把源代码文件通过diff的方式发给Linus，然后由Linus本人通过手工方式合并代码！

Linus Torvalds 在 2002 年起，使用 BitMover 的版本控制软件 BitKeeper 管理 Linux 核心开发，而因为 BitKeeper 除商业付费版本，仅提供可免费使用但不允许修改重新编译的精简版本，引起了开源社区的不满，如自由软件之父 Richard stallman 也敢严厉批评 Linux Torvalds 使用非自由软件开发 Linux 核心。

在 2005 年，Samba 档案服务器开发人 Andrew Tridgell 写了连接 BitKeeper 存储库的简单程序，被 BitMover 创办人 Larry McVoy 指控对 BitKeeper 进行逆向工程，因为决定停止 BitKeeper 对 Linux 的支持。

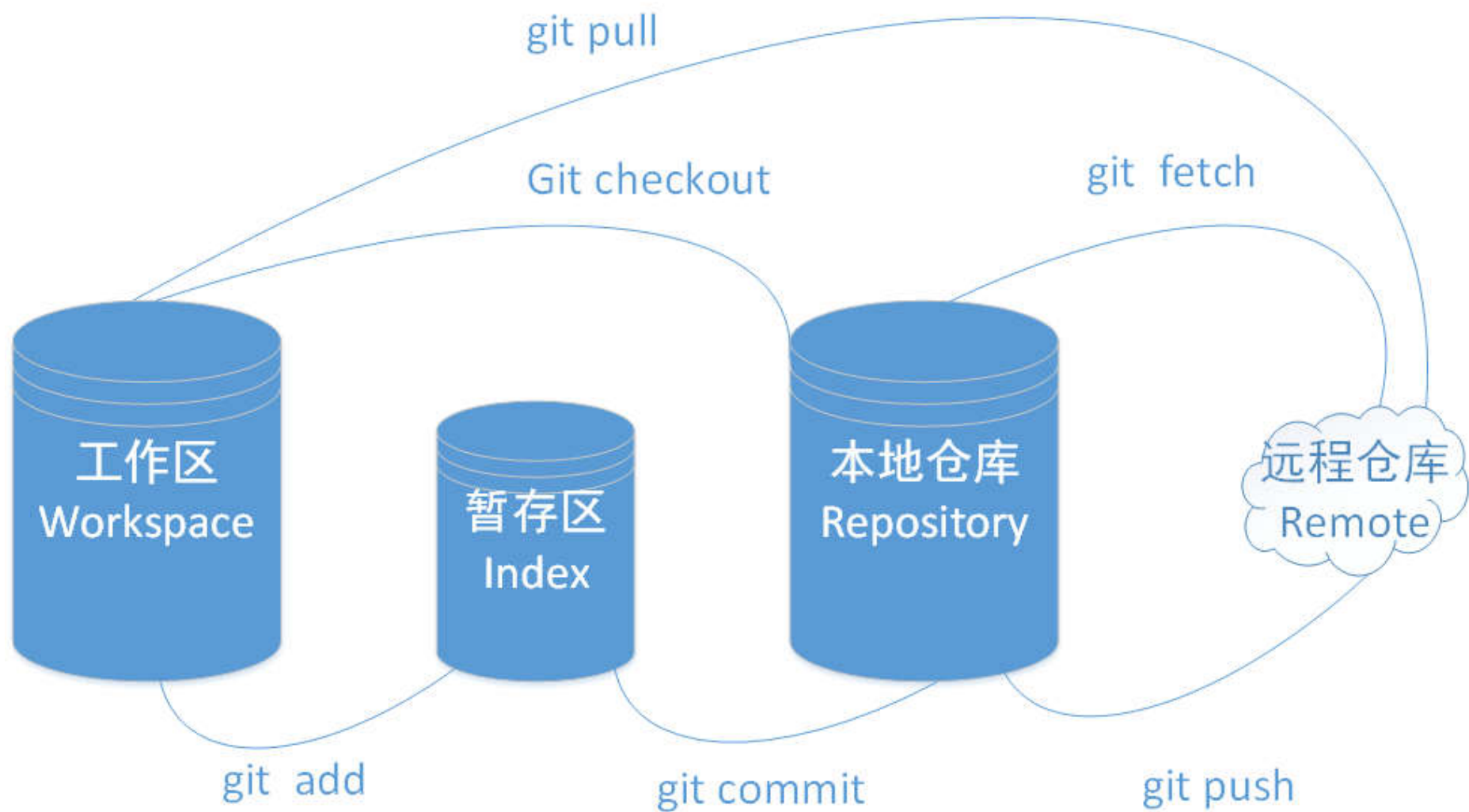
顿时 Linux 核心开发受到了严峻的挑战，而 Linus Torvalds 秉承自己的版本自己写的精神，整个周末都不见人影，隔周却如变戏法般带着 Git 出现。

## git和svn对比

1.git是分布式的，SVN不是 2.git把内容按元数据方式存储，而SVN是按文件 3.git分支和SVN的分支不同 4.git的内容完整性要优于SVN

## git原理

---



Itcast python

# git命令大全

---

## 新建代码仓库

---

# 在当前目录新建一个Git代码库

```
$ git init
```

# 新建一个目录，将其初始化为Git代码库

```
$ git init [project-name]
```

# 下载一个项目和它的整个代码历史

```
$ git clone [url]
```

## 配置

---

Git的设置文件为.gitconfig，它可以在用户主目录下（全局配置），也可以在项目目录下（项目配置）。

# 显示当前的Git配置

```
$ git config --list
```

# 编辑Git配置文件

```
$ git config -e [--global]
```

# 设置提交代码时的用户信息

```
$ git config [--global] user.name "[name]"
```

```
$ git config [--global] user.email "[email address]"
```

## 增加/删除文件

---

# 添加指定文件到暂存区

```
$ git add [file1] [file2] ...
```

# 添加指定目录到暂存区，包括子目录

```
$ git add [dir]
```

# 添加当前目录的所有文件到暂存区

```
$ git add .
```

# 删除工作区文件，并且将这次删除放入暂存区

```
$ git rm [file1] [file2] ...
```

# 改名文件，并且将这个改名放入暂存区

```
$ git mv [file-original] [file-renamed]
```

## 代码提交

---

# 提交暂存区到仓库区

```
$ git commit -m [message]
```

# 提交暂存区的指定文件到仓库区

```
$ git commit [file1] [file2] ... -m [message]
```

# 提交工作区自上次`commit`之后的变化，直接到仓库区

```
$ git commit -a
```

# 使用一次新的`commit`，替代上一次提交

# 如果代码没有任何新变化，则用来改写上一次`commit`的提交信息

```
$ git commit --amend -m [message]
```

```
# 重做上一次commit，并包括指定文件的新变化
$ git commit --amend [file1] [file2] ...
```

## 分支

---

```
# 列出所有本地分支
$ git branch
```

```
# 列出所有远程分支
$ git branch -r
```

```
# 列出所有本地分支和远程分支
$ git branch -a
```

```
# 新建一个分支，但依然停留在当前分支
$ git branch [branch-name]
```

```
# 新建一个分支，并切换到该分支
$ git checkout -b [branch]
```

```
# 新建一个分支，指向指定commit
$ git branch [branch] [commit]
```

```
# 新建一个远程分支，本地新建然后推送
$ git push origin [branch]
```

```
# 新建一个分支，与指定的远程分支建立追踪关系
$ git branch --track [branch] [remote-branch]
```

```
# 切换到指定分支，并更新工作区
$ git checkout [branch-name]
```

```
# 切换到上一个分支
$ git checkout -

# 建立追踪关系，设置当前分支与指定的远程分支之间关联
$ git branch --set-upstream-to [remote-branch]

# 合并指定分支到当前分支
$ git merge [branch]

# 选择一个commit，合并进当前分支
$ git cherry-pick [commit]

# 删除分支
$ git branch -d [branch-name]

# 删除远程分支
$ git push origin --delete [branch-name]
$ git branch -dr [remote/branch]
```

## 标签

---

```
# 列出所有tag
$ git tag

# 新建一个tag在当前commit
$ git tag [tag]

# 新建一个tag在指定commit
$ git tag [tag] [commit]

# 删除本地tag
$ git tag -d [tag]
```

```
# 删除远程tag  
$ git push origin :refs/tags/[tagName]
```

```
# 查看tag信息  
$ git show [tag]
```

```
# 提交指定tag  
$ git push [remote] [tag]
```

```
# 提交所有tag  
$ git push [remote] --tags
```

```
# 新建一个分支，指向某个tag  
$ git checkout -b [branch] [tag]
```

## 查看信息

---

```
# 显示有变更的文件  
$ git status
```

```
# 显示当前分支的版本历史  
$ git log
```

```
# 显示commit历史，以及每次commit发生变更的文件  
$ git log --stat
```

```
# 搜索提交历史，根据关键词  
$ git log -S [keyword]
```

```
# 显示某个commit之后的所有变动，每个commit占据一行  
$ git log [tag] HEAD --pretty=format:%s
```

```
# 显示某个commit之后的所有变动，其"提交说明"必须符合搜索条件
```



```
$ git log [tag] HEAD --grep feature
```

```
# 显示某个文件的版本历史，包括文件改名
```

```
$ git log --follow [file]
```

```
$ git whatchanged [file]
```

```
# 显示指定文件相关的每一次diff
```

```
$ git log -p [file]
```

```
# 显示过去5次提交
```

```
$ git log -5 --pretty --oneline
```

```
# 显示所有提交过的用户，按提交次数排序
```

```
$ git shortlog -sn
```

```
# 显示指定文件是什么人在什么时间修改过
```

```
$ git blame [file]
```

```
# 显示暂存区和工作区的差异
```

```
$ git diff
```

```
# 显示暂存区和上一个commit的差异
```

```
$ git diff --cached [file]
```

```
# 显示工作区与当前分支最新commit之间的差异
```

```
$ git diff HEAD
```

```
# 显示两次提交之间的差异
```

```
$ git diff [first-branch]...[second-branch]
```

```
# 显示某次提交的元数据和内容变化
```

```
$ git show [commit]
```

```
# 显示某次提交发生变化的文件
```

```
$ git show --name-only [commit]
```

# 显示某次提交时，某个文件的内容  
`$ git show [commit]:[filename]`

# 显示当前分支的最近几次提交  
`$ git reflog`

## 远程同步

---

# 下载远程仓库的所有变动  
`$ git fetch [remote]`

# 显示所有远程仓库  
`$ git remote -v`

# 显示某个远程仓库的信息  
`$ git remote show [remote]`

# 增加一个新的远程仓库，并命名  
`$ git remote add [shortname] [url]`

# 取回远程仓库的变化，并与本地分支合并  
`$ git pull [remote] [branch]`

# 上传本地指定分支到远程仓库  
`$ git push [remote] [branch]`

# 强行推送当前分支到远程仓库，即使有冲突  
`$ git push [remote] --force`

# 推送所有分支到远程仓库  
`$ git push [remote] --all`

# 撤销

---

# 恢复暂存区的指定文件到工作区

```
$ git checkout [file]
```

# 恢复某个`commit`的指定文件到暂存区和工作区

```
$ git checkout [commit] [file]
```

# 恢复暂存区的所有文件到工作区

```
$ git checkout .
```

# 重置暂存区的指定文件，与上一次`commit`保持一致，但工作区不变

```
$ git reset [file]
```

# 重置暂存区与工作区，与上一次`commit`保持一致

```
$ git reset --hard
```

# 重置当前分支的指针为指定`commit`，同时重置暂存区，但工作区不变

```
$ git reset [commit]
```

# 重置当前分支的`HEAD`为指定`commit`，同时重置暂存区和工作区，与指定`commit`一致

```
$ git reset --hard [commit]
```

# 重置当前`HEAD`为指定`commit`，但保持暂存区和工作区不变

```
$ git reset --keep [commit]
```

# 新建一个`commit`，用来撤销指定`commit`

# 后者的所有变化都将被前者抵消，并且应用到当前分支

```
$ git revert [commit]
```

# 暂时将未提交的变化移除，稍后再移入

```
$ git stash
```

```
$ git stash pop
```

## 其他

---

# 生成一个可供发布的压缩包

```
$ git archive
```

# 删除项目的`git`管理

```
$ rm .git
```

## github

---

GitHub于2008年上线，目前，除了Git代码仓库托管及基本的Web管理界面以外，还提供了订阅、讨论组、文本渲染、在线文件编辑器、协作图谱（报表）、代码片段分享（**Gist**）等功能。正因为这些功能所提供的便利，又经过长期的积累，GitHub的用户活跃度很高，在开源世界里享有深远的声望，形成了所谓的社交化编程文化（**Social Coding**）。

GitHub允许你免费创建不限数量的“公开”代码仓库，虽有总的资源限额，但一般情况下够用。如果不想公开代码，或者你的代码占用存储空间比较多，你也可以选择成为付费的会员。在GitHub，想要接触高水平的软件项目，了解最前沿的技术趋势，和上微博一样简单。

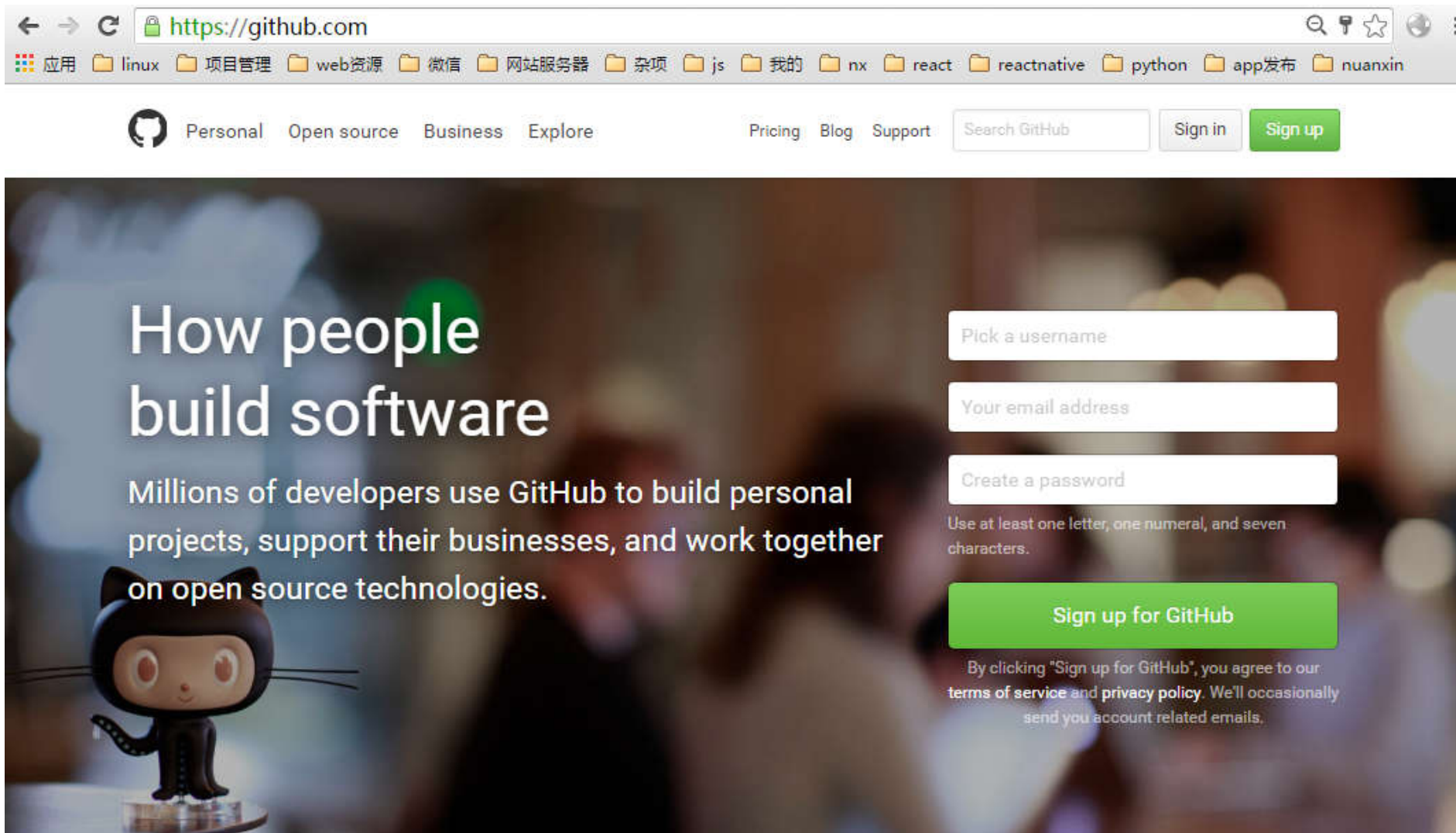
Github 在代码托管领域是先行者，现在强力的竞争对手也有不少，包括 Gitlab（局域网部署）、Bitbucket（免费账号不限 private 项目个数）、GitCafe（对国内开发者来说可能有墙内优势）。

## 注册

---

登陆github网站

<https://github.com>



创建一个项目

https://github.com/new

linux 项目管理 web资源 微信 网站服务器 杂项 js 我的 nx react reactnative python app发布 nuanxin

Search GitHub

Pull requests Issues Gist

+

New repository

Import repository


New organization

## Create a new repository

A repository contains all the files for your project, including the revision history.

Owner


Repository name

 xwpfullstack ▾


 /

Great repository names are short and memorable. Need inspiration? How about [scaling-happiness](#).

Description (optional)

☒  Public

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


☐  Private

You choose who can see and commit to this repository.

☒ Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** ▾

Add a license: **None** ▾ 

Create repository

- 创建一个python项目

https://github.com/xwpfullstack/python

项目目录: linux, 项目管理, web资源, 微信, 网站服务器, 杂项, js, 我的, nx, react, reactnative, python, app发布, nuanxin

GitHub navigation: This repository, Search, Pull requests, Issues, Gist

Repository: xwpfullstack / python

Actions: Unwatch (1), Star (0), Fork (0)

Tabs: Code, Issues (0), Pull requests (0), Projects (0), Wiki, Pulse, Graphs, Settings

传智python — Edit

Repository stats: 1 commit, 1 branch, 0 releases, 1 contributor

Branch: master | New pull request

Buttons: Create new file, Upload files, Find file, Clone or download

Commit history: xwpfullstack Initial commit

Files: README.md (Initial commit)

Clone with HTTPS (Use SSH)

Use Git or checkout with SVN using the web URL.

https://github.com/xwpfullstack/pythor

Open in Desktop | Download ZIP

python

传智python

- clone项目

```
python@ubuntu:~/workspace/teach$ git clone https://github.com/xwpfullstack/python.git
正克隆到 'python'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
展开对象中: 100% (3/3), 完成.
检查连接... 完成。
python@ubuntu:~/workspace/teach$ ls
python
python@ubuntu:~/workspace/teach$ cd python/
python@ubuntu:~/workspace/teach/python$ ls
README.md
python@ubuntu:~/workspace/teach/python$
```

- 第一次使用需配置用户名和邮箱

```
$ git config --global user.name "xxx"
$ git config --global user.email "xxx@itcast.com"
```

- 完成第一次提交代码



```
python@ubuntu:~/workspace/teach/python$ ls
README.md
python@ubuntu:~/workspace/teach/python$ echo "print('python is so magic')" > test.py
python@ubuntu:~/workspace/teach/python$ ls
README.md  test.py
python@ubuntu:~/workspace/teach/python$ git add test.py
python@ubuntu:~/workspace/teach/python$ git commit -m "init test.py"
[master 8525ab3] init test.py
 1 file changed, 1 insertion(+)
 create mode 100644 test.py
python@ubuntu:~/workspace/teach/python$ git push origin master
Username for 'https://github.com': xwpfullstack
Password for 'https://xwpfullstack@github.com':
对象计数中: 3, 完成.
压缩对象中: 100% (2/2), 完成.
写入对象中: 100% (3/3), 301 bytes | 0 bytes/s, 完成.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/xwpfullstack/python.git
 76f03ee..8525ab3  master -> master
python@ubuntu:~/workspace/teach/python$
```

- 刷新查看github上的对应项目

https://github.com/xwpfullstack/python

linux 项目管理 web资源 微信 网站服务器 杂项 js 我的 nx react reactnative python app发布 nuanxi

This repository Search

Pull requests Issues Gist

xwpfullstack / python

Unwatch 1

Star 0

Fork 0

Code

Issues 0

Pull requests 0

Projects 0

Wiki

Pulse

Graphs

Settings

传智python — Edit

2 commits 1 branch 0 releases 1 contributor

Branch: master New pull request

Create new file Upload files Find file Clone or download

xwpfullstack init test.py Latest commit 8525ab3 a minute ago

README.md Initial commit 16 minutes ago

test.py init test.py a minute ago

README.md

python

传智python

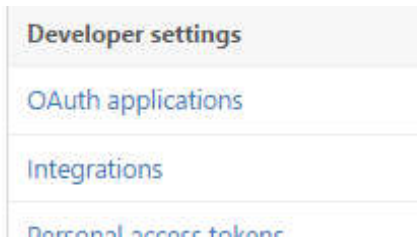
# ssh配置

- 可以为项目配置身份密钥，这样不必每次推送都输入账号和密码

The screenshot shows the GitHub 'Settings/keys' page. The browser address bar displays 'https://github.com/settings/keys'. The top navigation bar includes links for 'Pull requests', 'Issues', and 'Gist'. On the left sidebar, 'SSH and GPG keys' is highlighted with a red box. The main content area is titled 'SSH keys' and contains a list of three SSH keys:

Key Name	Fingerprint	Added On	Last Used	Action
GitHub for Mac [redacted] MacBook Air	cb:52:7b:83:43:59:df:62:9c:2c:7e:6d:65:3c:17:b6	8 Aug 2015	within the last 13 months	
ubuntu16.04	b6:46:7c:a3:9f:84:d5:9c:18:4f:7f:bd:bf:a1:87:d7	20 May 2016	within the last 3 weeks	
ubuntu14.04	36:d7:24:1b:0a:3d:30:16:8f:79:48:e0:7a:55:37:f5	31 Jul 2016	within the last 3 months	Delete

Below the list, there is a link to 'Check out our guide to generating SSH keys or troubleshoot common SSH Problems.' At the bottom, the 'GPG keys' section is visible, showing 'There are no GPG keys with access to your account.' and a 'New GPG key' button. A user dropdown menu is open on the right, showing 'Signed in as xwpfullstack' and a list of links including 'Settings', which is highlighted with a red box.



- ubuntu下生成当前电脑ssh秘钥

# 一路敲回车

\$ ssh-keygen

```
python@ubuntu:~/workspace/teach/python$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/python/.ssh/id_rsa):
/home/python/.ssh/id_rsa already exists.
Overwrite (y/n)?
python@ubuntu:~/workspace/teach/python$
```

```
python@ubuntu:~/ssh$ pwd
/home/python/.ssh
python@ubuntu:~/ssh$ ls
id_rsa  id_rsa.pub  known_hosts
python@ubuntu:~/ssh$
```

id\_rsa是私钥，自己保管好  
id\_rsa.pub是公钥，上传至github

- 把公钥上传至github作为ssh keys





Personal settings

Profile

Account

Emails

Notifications

Billing

SSH and GPG keys

Security

Blocked users

Repositories

Organizations

Saved replies

Authorized applications

Installed integrations

Developer settings

OAuth applications

Integrations

Personal access tokens

Organization settings

SSH keys

New SSH key

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

GitHub for Mac - 邢文鹏的MacBook Air

Fingerprint: cb:52:7b:83:43:59:df:62:9c:2c:7e:6d:65:3c:17:b6

Added on 8 Aug 2015 by GitHub for Mac — Last used within the last 13 months

Delete

ubuntu16.04

Fingerprint: b6:46:7c:a3:9f:84:d5:9c:18:4f:7f:bd:bf:a1:87:d7

Added on 20 May 2016 — Last used within the last 3 weeks

Delete

ubuntu14.04

Fingerprint: 36:d7:24:1b:0a:3d:30:16:8f:79:48:e0:7a:55:37:f5

Added on 31 Jul 2016 — Last used within the last 3 months

Delete

Title

python-ubuntu16.04

Key

ssh-rsa  
AAAAB3NzaC1yc2EAAAADAQABAAQDM8fYqwr44HI/biK6luwqU+Md5YuGsK3mIMmgMZFMp7lBeJOLBpSd  
YcD/LgafcQ3bE6Ae2IMNpEV+3LyBcdQl6hEzR/8/kdjV6y5cXkBjy1v5EDen0NSiswBVHEf2tP0ttYZKk1/R3V49dTZTY  
AOYCaps4OIzIWp2G/f+iTMHcJEsBJ6DirY5synfLwEzo3eliwnNXscHn+D1leofq6pv2L4MJ6v/gX7nLmtbSbLR78MA  
uSwyigna5UJ3FOpnvzLNg0D0OYNYDEloE1SwH0gcrwQzOJ2Ujmvkh9L54vnQo5n85oziy+Xg78FLHxAQSrgTPnBh8  
dcs8UvbsCaW9tp8np python@ubuntu

Add SSH key



- 修改项目的配置文件为ssh方式,编辑项目里的.git/config文件

```
1 [core]
2     repositoryformatversion = 0
3     filemode = true
4     bare = false
5     logallrefupdates = true
6 [remote "origin"]
7     #url = https://github.com/xwpfullstack/python.git
8     url = git@github.com:xwpfullstack/python.git
9     fetch = +refs/heads/*:refs/remotes/origin/*
10 [branch "master"]
11     remote = origin
12     merge = refs/heads/master
```

- 修改项目代码，再次提交，无须输入账号和密码
- 协作，添加可对项目进行

https://github.com/xwpfullstack/python/settings/collaboration

linux 项目管理 web资源 微信 网站服务器 杂项 js 我的 nx react reactnative python app发布 nuan

This repository Search Pull requests Issues Gist

xwpfullstack / python Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Pulse Graphs Settings

Options

**Collaborators**

Branches

Webhooks

Integrations & services

Deploy keys

Collaborators Push access to the repository

This repository doesn't have any collaborators yet. Use the form below to add a collaborator.

Search by username, full name or email address

You'll only be able to find a GitHub user by their email address if they've chosen to list it publicly. Otherwise, use their username instead.

Add collaborator

- 也可以建立一个team 把你的伙伴加到team 然后把项目放到team下

https://github.com/organizations/new

linux 项目管理 web资源 微信 网站服务器 杂项 js 我的 nx react reactnative python app发布 nua

Search GitHub Pull requests Issues Gist

Create an organization

New repository

Import repository

# Create an organization

[New organization](#)

Completed  
Set up a personal account



Step 2:  
Set up the organization



Step 3:  
Invite members

## Set up the organization

Organization name

The organization will live at <https://github.com/>

Billing email

Receipts will be sent here

### Plan

- ☒ Unlimited members and public repositories for free.
- ☐ Unlimited private repositories at \$25/month for your first 5 users, \$9/month for each additional user.

[Create organization](#)

## Organizations

- ✓ Repository management
- ✓ Fine-grained permissions
- ✓ Focused dashboard

The credit card and plan you choose on this screen will be billed to the organization — not your user account (xwpfullstack).

[Learn more](#)

发现优秀项目



# 私有git服务器解决方案

---

- gitlab
- gitolite

## 分支设计

---

master、develop

## 场景

---

小明笨笨志强

## 实验

---