Git

讲师：pansir

简介

安装

ubuntu：`sudo apt-get install git`

macos：`brew install git`

windows：手动下载并安装：https://git-scm.com/download/win。完成后 开始->Git->Git Bash打开命令行即可

- 设置全局配置

$ git config -global user.name "pansir"

$ git config -global user.name "xxx@qq.com"

版本库

可以理解成一个目录，目录里的所有文件都可以被Git管理，包括：文件的修改，删除，记录每一次改动，可以回退

1. 创建一个空目录：

```
$ cd ~/mygit/github_project
$ mkdir first_git
$ cd first_git
```

2. 初始化仓库：`$ git init`
3. 查看隐藏目录：`$ ls -a`

Git管理

改动说明

Git只跟踪文本文件的改动 Tips：不要用windows自带的记事本，建议使用Notepad++编辑，注意Notepad++的默认编码设置为UTF-8 without BOM

初次添加

1. 创建一个新文件：vim test.txt，输入：`hello git`，保存
2. 添加到暂存区：git add test.txt 没有提示，则说明成功

3. 提交修改：git commit -m "add test.txt"
4. git add 可以添加多个文件，最后一次性提交
5. 必须指定注释，而且一定要规范

修改内容

1. 修改test.txt内容
2. 查看状态：git status
3. 查看具体的改动：git diff test.txt
4. 提交到暂存区：git add test.txt
5. 再次查看状态：git status
6. 提交内容：git commit -m "add text"
7. 再次查看状态：git status

版本回退

1. 再次重复上面步骤，修改文件后提交
2. 查看版本信息：git log，可以看到每次的commit记录
3. 版本标识：
    - HEAD：当前版本
    - HEAD^：上个版本
    - HEAD^^：上上个版本
    - HEAD～100：往上100个版本
4. 回退版本：git reset --hard HEAD^
5. 查看文件，发现已经还原
6. 往上找到版本号，再往未来跳转：git reset --hard 7a923b2d397d1aab13a8924e7596949f77692e32（最近的版本ID）
7. 查看文件与状态：git log，发现又前进了

工作区/暂存区

工作区：即工作目录 版本库：即隐藏的.git，版本库里包含了暂存区(stage/index)，master分支，即HEAD指针(指向master)

1. git add xxx：把文件添加到暂存区
2. git commit：把暂存区内容提交到当前分支

暂存区演示：

1. git status：工作区干净
2. 修改文件 + 添加一个文件，git status
3. 添加到暂存区：git add test.txt + git add a.text，git status
4. 提交所有修改：git commit -m "add file"，git status

撤销改动

1. 丢弃工作区某个文件改动，`git checkout -- <file>`

2. 改动工作区内容，并提交到了暂存区，需要丢弃暂存区改动
    - 首先：`git reset HEAD <file>` 撤销提交暂存区，回到第一步
    - 然后：`git check out -- <file>`
3. 如果提交到了本地分支，需要先进行版本回退

删除文件

方式1：

1. 磁盘删除：rm a.txt
2. 版本库删除：git rm a.txt 或者 git add a.txt
3. 提交修改：git commit

方式2：

1. 版本库直接删除：git rm a.txt 或者 git add a.txt
2. 提交修改：git commit

删除回退

1. 丢弃工作区某个文件删除改动，`git checkout -- <file>`
2. git rm 后会直接提交到暂存区，所以需要丢弃暂存区改动
    - 首先：`git reset HEAD <file>` 撤销提交暂存区，回到第一步
    - 然后：`git check out -- <file>`
3. 如果提交到了本地分支，需要先进行版本回退

建立远程连接

1. 创建ssh-key： cd ~ ssh-keygen -t rsa -C "xxx@xxx.com"
2. 一路回车，使用默认值
3. 检查home目录：cd ~/.ssh 发现id_rsa和ir_rsa.pub文件
4. 登陆github，头像->Settings->SSH and GPG keys -> new SSH key
5. title随便填，把公钥复制进来

添加远程仓库

1. "+"号 -> New respository -> 确认
2. 点击ssh，复制地址，关联远程仓库： $ git remote add origin git@github.com:xiaopanddxiong/first_git.git
3. 查看远程分支：git remote -vv
4. 推到远程分支：git push -u origin master

日常操作

1. 更改文件内容：vim test.txt
2. 提交到暂存区：git add .
3. 提交到仓库：git commit -m "xxx"
4. 推到远程分支：git push origin master

拉取新的远程仓库

1. 创建新的远程仓库："+"号 -> New respository -> 勾选 Initialze ... -> 确认
2. 拉取到本地：git clone git@github.com:xiaopanddxiong/second_git.git

分支管理

创建与合并分支

HEAD指向master，master指向最新的提交 过程演示：

1. HEAD -> master -> 最新
2. 新增并修改：HEAD -> dev -> 最新
3. 修改dev分支内容，dev前进，但master不变
4. 修改指针：HEAD -> master -> (dev)最新
5. 删除指针：dev->最新

命令演示：

1. 创建新分支：git checkout -b dev
2. 查看分支：git branch -a
3. 新增一个文件：vim say.py
4. 提交：git add . git commit -m "add say"
5. 切换到master：git checkout master
6. 将dev合并到master分支：git merge dev
7. 删除dev分支：git branch -d dev
8. 查看分支：git branch -a

解决冲突

1. 创建一个新分支：git checkout -b dev
2. 修改readme内容：ok dev
3. 提交：git add . git commit -m "add dev"
4. 切换到master：git checkout master
5. 修改master下的readme：ok master
6. 提交：git add . git commit -m "add master"
7. 合并分支：git merge dev 提示冲突
8. 打开文件：vim readme.md

```
<<<<<<< HEAD
ok dev
=======
ok master
>>>>>>> feature1
```

9. 手动解决冲突，改成我们希望的样子，然后保存
10. 再次提交：git add . git commit -m "conflict"
11. 查看分支合并情况：git log --graph --pretty=oneline --abbrev-commit

12. 删除dev分支：git branch -d

补充命令

- 查看配置：git config -l
- 查看所有本地分支：git branch -vv
- 查看所有关联远程分支：git remote -vv