**Git笔记**

### 1. 工作区（Working Directory）和版本库（Repository）

[](https://github.com/michaelliao/learngit/blob/master/Git%E5%AD%A6%E4%B9%A0%E7%AC%94%E8%AE%B0/git.jpeg)

**说明：**

* 工作区就是创建仓库的文件夹如（learngit文件夹就是一个工作区）
* 版本库就是工作区的隐藏目录.git,版本库中有暂存区（stage/index）和分支（master）
* git add 实际是把文件添加到暂存区， git commit 把暂存区的内容提交到当前分支

### 2.创建版本库

1. 创建git仓库文件夹，名为：learngit
2. $ mkdir learngit
3. 进入leadngit文件夹
4. $ cd learngit
5. 初始化git仓库
6. $ git init

### 3. 添加文件

1. 在leangit下添加一个readme.txt文件，并编辑一些内容
2. 添加到仓库暂存区（）在暂存区 文件会变绿
3. $ git add readme.txt
4. 提交readme.txt文件到当前分支, -m "提交说明"(只有进行 git add 后 go commit 命令才有效)
5. $ git commit -m "add readme.txt"

### 4. 修改文件

#### 4.1 当文件在工作区时

1. 查看readme.txt文件内容
2. $ cat readme.txt
3. 修改readme.txt文件内容
4. 查看仓库状态
5. $ git status
6. 添加到仓库暂存区，并提交到分支
7. $ git add readme.txt
8. $ git commit -m "modify readme.txt"

#### 4.2 当文件在暂存区时

1. 修改文件内容
2. 添加到仓库暂存区
3. $ git add readme.txt
4. 提交到分支
5. $ git commit -m "modify readme.txt at the stage"

### 5. 撤销修改文件（未提交到分支）

#### 5.1 当文件在工作区时

1. 执行撤销命令
2. $ git checkout -- readme.txt

#### 5.2 当文件在暂存区时

1. 令文件回到工作区
2. $ git reset HEAD readme.txt
3. 执行撤销命令
4. $ git checkout -- readme.txt

### 6. 版本控制（无限次后悔）

说明：在Git中，HEAD表示当前版本，HEAD^表示上一版本 HEAD^^表示上上一个版本

1. 查看提交日志输出(完整版)
2. $ git log
3. 查看提交日志输出（精简版）
4. $ git log --pretty=noline
5. 回到上一版本
6. $ git reset --hard HEAD^
7. 回到指定版本（hard 后面添加版本号）
8. $ git reset --hard ea34578
9. 查看命令历史
10. $ git reflog

### 7. 远程仓库（github）

#### 7.1 添加到远程库

1. 在github上创建一个名为learngit的空仓库
2. 在本地learngit仓库下运行命令
3. $ git remote add origin git@github.com:iphone5solo/learngit.git
4. 把本地内容推送到github远程库上(第一次push 参数带 -u 关联远程仓库)
5. $ git push -u origin master

注意：如果在git push -u origin master时出现以下错误，证明电脑没有修改远程仓库的公钥，

Permission denied (publickey).

fatal: Could not read from remote repository.

Please make sure you have the correct access rights

and the repository exists.

解决方法：

1. 在github上点击Edit profile --> SSH and GPG keys --> new SSH key 添加SHH公钥
2. 打开id\_rsa.pub文件（/Users/iphone5solo/.ssh/id.rsa.pub）
3. 将id\_rsa.pub文件内容拷贝到key就可以了，title随便填。

#### 7.2 从远程库克隆

1. 在github上创建一个名为clonegit的仓库
2. 使用命令克隆仓库
3. $ git clone git@github.com:iphone5solo/clonegit

#### 7.3 从远程仓库更新本地仓库（已关联）

$ git pull origin master

### 8. 分支管理

**图文说明：**

master分支是一条线，git用master指向最新的提交，在用HEAD指向master，以此才确定当前分支，和提交点。

[](https://github.com/michaelliao/learngit/blob/master/Git%E5%AD%A6%E4%B9%A0%E7%AC%94%E8%AE%B0/master.png)

**1. 创建分支**

当我们创建新的分支，例如newBranch,git就会新建一个指针newBranch,指向master相同的提交，再把HEAD指向newBranch,就表示当前分支在newBrach上

[](https://github.com/michaelliao/learngit/blob/master/Git%E5%AD%A6%E4%B9%A0%E7%AC%94%E8%AE%B0/newBranch.png)

**2. 新分支的修改和提交**

现在对工作区的修改和提交就是针对newBranch分支了，比如新提交一次后，newBranch指针就向前移动一步，指向最新提交，而master指针指向不变。

[](https://github.com/michaelliao/learngit/blob/master/Git%E5%AD%A6%E4%B9%A0%E7%AC%94%E8%AE%B0/commitAtNewBranch.png)

**3. 分支的合并**

把newBranch合并到master上，直接把master指向newBranch的当前提交，就完成了合并。

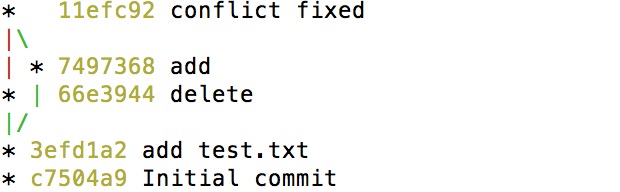
[](https://github.com/michaelliao/learngit/blob/master/Git%E5%AD%A6%E4%B9%A0%E7%AC%94%E8%AE%B0/merge.png)

**4. 合并完成删除分支**

删除分支newBranch，就剩下一个master分支

[](https://github.com/michaelliao/learngit/blob/master/Git%E5%AD%A6%E4%B9%A0%E7%AC%94%E8%AE%B0/masterBranch.png)

**5. 查看分支合并图解**

[](https://github.com/michaelliao/learngit/blob/master/Git%E5%AD%A6%E4%B9%A0%E7%AC%94%E8%AE%B0/branch.png)

**关于分支的主要命令如下**

* 查看分支
* $ git branch
* 创建newBranch分支
* $ git branch newBranch
* 切换HEAD指向newBranch分支
* $ git checkout newBranch
* 创建+切换分支
* $ git checkout -b newBranck
* 合并某分支到当前分支
* $ git merge newBranch
* 普通删除newBranch分支
* $ git branch -d newBranch
* 强行删除newBranch分支
* $ git branch -D newBranch
* 查看分支合并状况
* $ git log --graph --pretty=oneline --abbrev-commit

### 9. 藏匿当前未提交的分支

如： 当前在修改自己的分支dev,突然项目经理要求修复一个bug-07

解决方法：

1. 藏匿当前dev分支的工作状态
2. $ git stash
3. 新建一个bug-07分支
4. $ git branch -b bug-07
5. 修复bug并提交，合并bug-07到master分支
6. $ git commit -m "fix the bug-07"
7. $ git checkout master
8. $ git merge --no-ff -m "merge bug-07" bug-07
9. 删除bug-07分支
10. $ git branch -d bug-07
11. 查看当前stash
12. $ git stash list
13. 恢复dev分支的工作状态，并删除stash内容
14. $ git stash pop

### 10. 多人协作

* 查看远程库信息
* $ git remote
  + 详细查看远程信息

$ git remote -v

* + 推送分支到远程库

$ git remote origin master

* + 抓取远程分支

$ git pull origin master

### 11. 标签管理

* 创建一个标签，默认为HEAD当前分支添加标签
* $ git tag v1.0
* 为版本号为e8b8ef6添加v2.0标签
* $ git tag v2.0 e8b8ef6
* 为版本号为6cb5a9e添加带有说明的标签，-a指定标签名,-m指定说明文字
* $ git tag -a v3.0 -m "version 0.2 released" 6cb5a9e
* 根据标签查看指定分支
* $ git show v0.2
* 查看所有标签
* $ git tag
* 删除v1.0标签
* $ git tag -d v1.0
* 把v0.9标签推送到远程
* $ git push origin v0.9
* 推送所有尚未推送到远程的本地标签
* $ git push origin --tags
* 删除远程标签, 先删除本地标签，再删除远程标签
* $ git tag -d v0.9

$ git push origin :refs/tags/v0.9