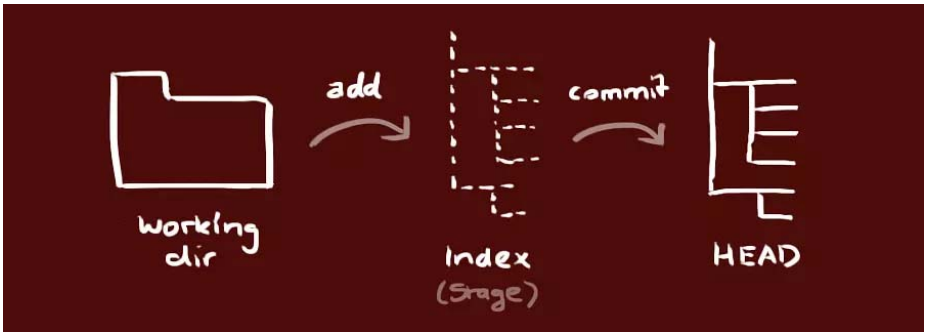
多用 Git 少交税

Kidult 2018-01-08 18:03:57

Git是一套版本管理系统。看到“Git版本管理”，一大部分盆友已经转身想走，在你握着门把手准备开门走人时，请最后听我说完最后一句：人人都需要版本管理，git可以帮你少交停电/蓝屏/死机税，提高产出效率，不来一发吗?



一、为什么要了解Git

试过半夜写汇报ppt吗?

'汇报ppt'→'汇报ppt1'→'汇报ppt11'→'汇报ppt2015-03-17'→'汇报ppt2015-03-17新'→'汇报ppt2015-03-17新1'……

无休止的命名斗争，这就是自然而然的版本管理，只不过，没有好的工具，所以显得一团 mess。

无论学生党还是设计师（改20个版本后终于顺利用回第1版），无论公众号运营还是音乐人，都持续产出着自己的“半成品/作品”。99.999% 的作品都不可能一气呵成，比如这篇笔记的第一个 commit 版本，简直惨不忍睹。如果有版本管理意识，以及高效、方便的工具，生活也许可以简单许多，更不要说天有不测风云的停电忘保存、脑残删备份等等好事等着我们。

来吧，fork 有用有趣的东西，git 你应该在意的东西，日拱一卒，打造我们的作品。

二、Git主要概念

Git 实现在本地和远端进行版本管理。

1.工作空间

Git 有四个空间概念：工作目录(workspace)，暂存区(index)，本地仓库(local repository)，远程仓库(remote repository)

想象一下，我们开一个包子店(图片做的不是同一款包子，见谅)~

- 首先，得有一张**大桌子**用来和面、擀皮儿、包馅等等，这张桌子相当于workspace，随你折腾的地方，工作主要都在这里进行。

Workspace

- 然后，包好的包子们会放到一个**蒸笼**里，等待被蒸，这个蒸笼就是index暂存区。蒸笼用来放我们想保留的成品或半成品，至于选哪些卖出去，这是以后考虑的事情。

Index

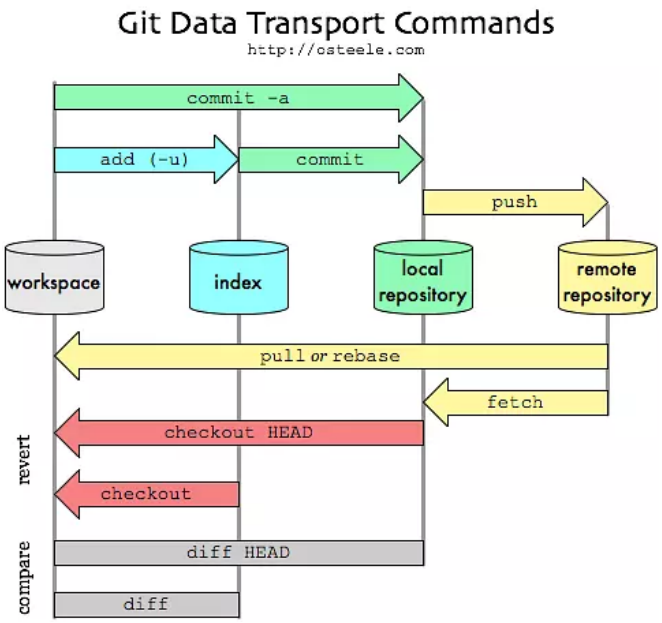
- 下一步，蒸包子。蒸好的包子已经可以吃了，但是我们还是得先把它们从蒸笼拿出来放在**盘子**里。盘子就类似本地仓库local repository，里面都是等待出货的好东西。当然，你也可以在最后一刻把看不顺眼的包子扔掉，或者自己吃掉。

Local repository

- 最后一步就是把包子送到**货架/客人的桌上**。公之于众的货架，就是远程仓库remote repository，丑媳妇终于见公婆啦。

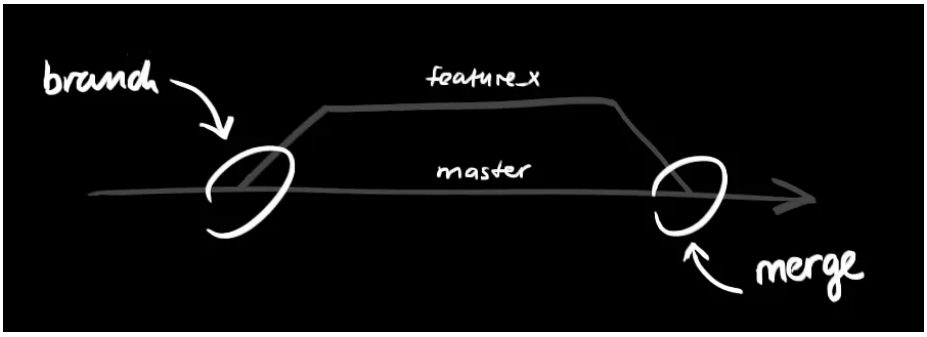
Remote repository

配合下面这张图，我们对Git就有一个基本概念了。



2.Head & branch & master & origin

Git 系统的实质更像是一棵大树，树干（就是 head 啦）是最后一次提交的成果。在树干上，你可以开无数的分支（就是 branch 啦）胡弄，弄乱了也不怕，大不了剪掉再开一个，树干不受任何影响。折腾ok的分支，最后可以 merge 到默认 branch 也就是 master 上。



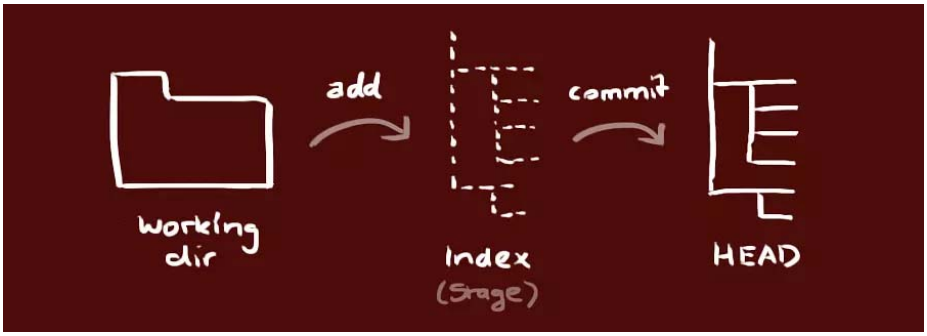
用技术性语言描述，分支用来将特性开发绝缘开来。在创建仓库的时候，master 是“默认的”分支。在其他分支上进行开发，完成后再将它们合并到主分支上。

那 origin 又是什么？origin 是远程默认的仓库。clone 完成之后，Git 会自动将远程仓库命名为 origin。

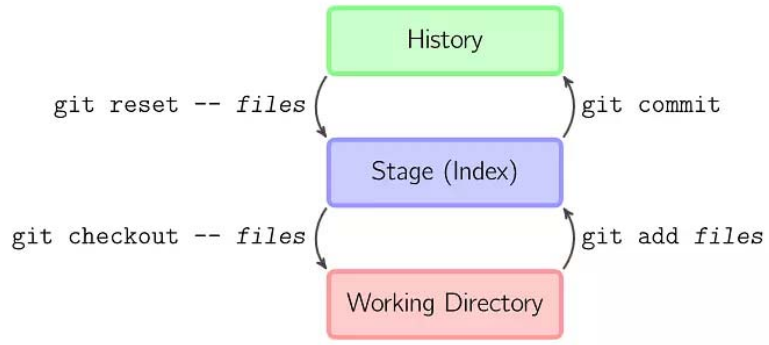
那 Head 和 master 又是什么关系？Head 其实只是个指针，指向当前最近 commit 的 branch。而 master 是本地默认的 branch，所以Head经常都是指向 master。另外 Head 是官方定义的，而 master和 origin 都是大家常用的命名，并不一定要叫 master 和 origin。[2]

3.工作流：add & commit & push

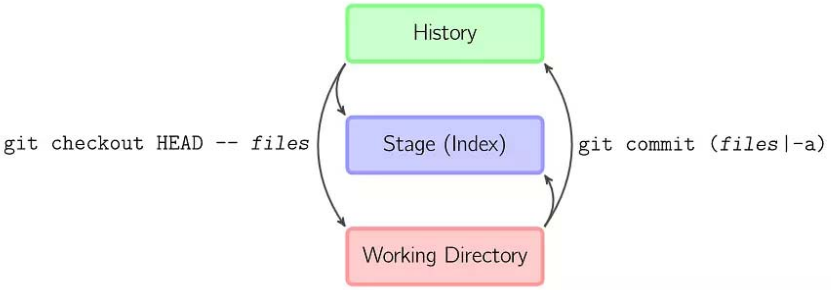
- 把包子从桌子挪到蒸笼，叫add————已修改modified————汇报ppt初稿写成；
- 把包子从蒸笼挪到盘子，叫commit————已暂存staged————汇报ppt完稿存到u盘/网盘什么的；
- 把包子从盘子挪到货架，叫push————已提交commited——汇报ppt发送到boss邮箱。



Git的好处之一是，包子包好后，还可以回退……



捷径：



三、配置

1.工作目录 2.本地仓库 3.远程仓库

四、常用命令

最常用：`git command --help`

1.创建

需要进入目标目录进行操作

- 创建新仓库：`git init`
- 创建一个本地仓库的克隆版本：`git clone /path/to/repository`
- 克隆远端服务器上的仓库： `git clone username@host:/path/to/repository`

2.查询

`git status`

- staged:已在index，等待被commit.
- unstaged:文件做了改动，但还不能被commit.
- untracked:Git还没有开始跟踪，需要先add.
- deleted:文件已被删除，等待remove.

Staging Area:commit前把文件们收集到一起，以便打包commit。

3.add/添加

- 添加到暂存区（让Git开始跟踪更改，也就是从untracked变为tracked）：`git add <filename>` 或 `git add *`
- 添加全部文件：`git add -A`， -A 表示包含删除的文件。
- git reset: `git reset <filename>` 从staging area移除文件。

4.commit/提交

"commit" 可以理解为一次快照，帮助我们把所有改动以timeline的方式组织起来。

- 提交改动(到head，但还没到远程服务器)：`git commit -m "代码提交信息"` git commit -m 'Add all files'
- 把所有当前目录下的文件加入暂存区域再运行commit：`git commit -a`

- 提交到远程仓库：`git push origin master` （可以把 master 换成你想要推送的任何分支）。 如果还没有克隆现有仓库，并想将仓库连接到某个远程服务器：`git remote add origin <server>`。

5.push/推送

将文件推送到远程仓库中：`git push -u origin master`。远程仓库默认叫**origin**。-u 告诉Git记住参数，下次可以直接使用push。

6.pull/拉取

更新本地仓库至最新改动：`git pull origin master`

7.checkout/切换

checkout命令用于从历史提交（或者暂存区域）中拷贝文件到工作目录，也可用于切换分支

- 切换分支： `git checkout <branch>`
- 新建并切换到分支：`git checkout -b new_branch` 等同于：`git branch new_branch` + `git checkout new_branch`
- 把文件从暂存区域复制到工作目录，用来丢弃本地修改：`git checkout --<files>`
- 回滚到复制最后一次提交:`git checkout HEAD -- <files>`

8.diff/比对

`git diff`

9.reset/撤销

- 从index中撤销所有文件：`git reset`
- 从index中撤销最后一次add的文件：`git reset --<flies>`
- 恢复之前版本：`git reset --hard`
- 回滚到最近一次：`git checkout -- <target>`

10.merge

合并其他分支到当前分支：`git merge`

11.remove & clean

- 从硬盘和index移除文件：`git rm`
- 删除分支`git branch -d <branch name>`

Ref

1. Git简明指南
2. What are the git concepts of HEAD, master, origin?
3. Try Git
4. 图解Git
5. Pro Git中文版
6. Gitmagic中文版

00 的其他文章

心智乐高04 - 很傻很天真的贝叶斯定理
学编程？从五个思维训练开始
拖延这件小事，讨论一次就够了
重启学习系统，做个知识炼金术士
知识炼金术士行动指南 1.0
把一个人活成一个公司，你可能就不会那么迷茫了