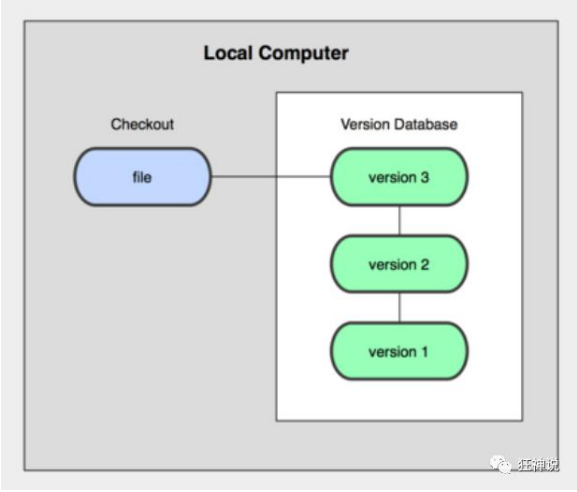
Git 的使用

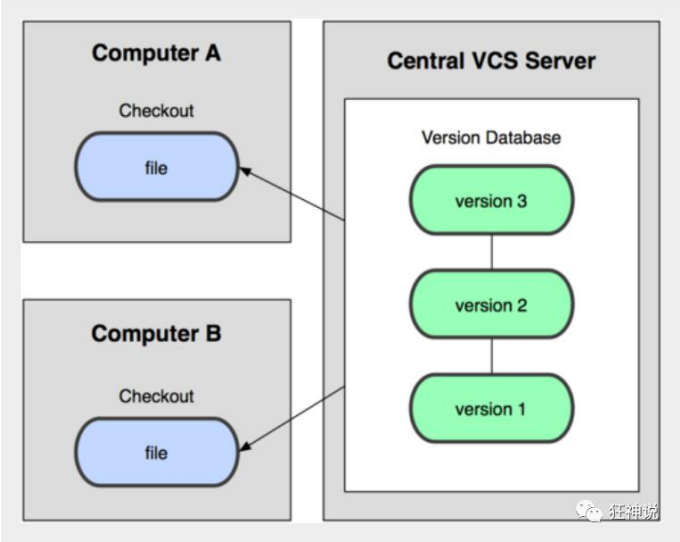
版本控制

版本控制（Revision control）是一种在开发的过程中用于管理我们对文件、目录或工程等内容的修改历史，方便查看更改历史记录，备份以便恢复以前的版本的软件工程技术。
实现跨区域多人协同开发
追踪和记载一个或者多个文件的历史记录
组织和保护你的源代码和文档
统计工作量
并行开发、提高开发效率
跟踪记录整个软件的开发过程
减轻开发人员的负担，节省时间，同时降低人为错误。
常用版本控制器
1.git
2.svn
3.cvs
4.vss
5.tfs
6.vs online

版本控制分类
1. 本地版本控制，适合个人使用



2. 集中版本控制，代表 svn，所有的版本数据保存到服务器上，协同开发者从服务器同步更新和上传自己的更新。

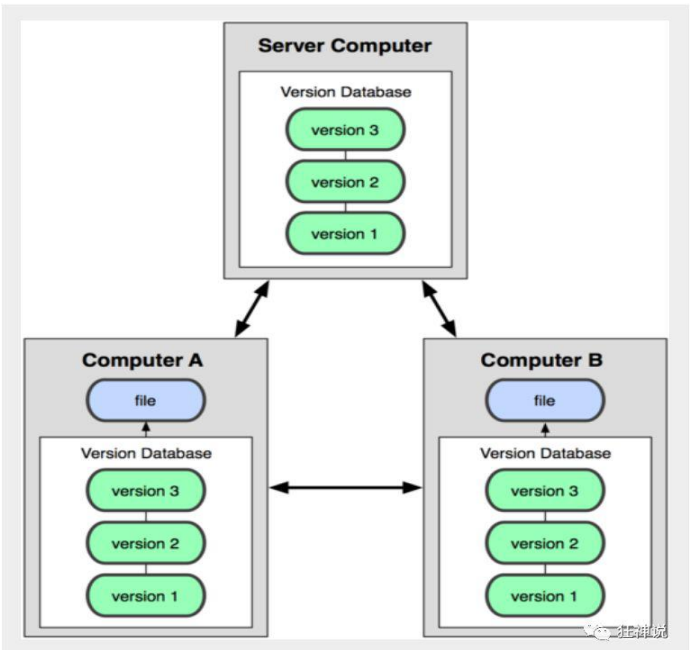


3.分布式版本控制 git

每个人都拥有全部的代码！安全隐患！

所有版本信息仓库全部同步到本地的每个用户，这样就可以在本地查看所有版本历史，可以离线在本地提交，只需在连网时 push 到相应的服务器或其他用户那里。由于每个用户那里保存的都是所有的版本数据，只要有一个用户的设备没有问题就可以恢复所有的数据，但这增加了本地存储空间的占用。

不会因为服务器损坏或者网络问题，造成不能工作的情况！



Git 和 SVN 的区别

SVN 是集中式版本控制系统，版本库是集中放在中央服务器的，而工作的时候，用的都是自己的电脑，所以首先要从中央服务器得到最新的版本，然后工作，

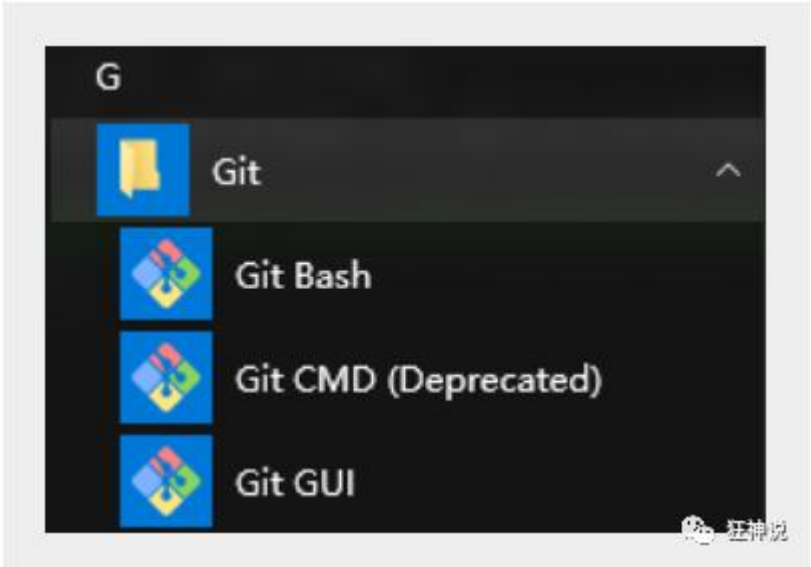
完成工作后，需要把自己做完的活推送到中央服务器。集中式版本控制系统是必须联网才能工作，对网络带宽要求较高。

Git 是分布式版本控制系统，没有中央服务器，每个人的电脑就是一个完整的版本库，工作的时候不需要联网了，因为版本都在自己电脑上。协同的方法是这样的：比如说自己在电脑上改了文件 A，其他人也在电脑上改了文件 A，这时，你们两之间只需把各自的修改推送给对方，就可以互相看到对方的修改了。Git 可以直接看到更新了哪些代码和文件！

Git 是世界上最先进的分布式版本控制系统。

Git 安装和环境配置

https://mp.weixin.qq.com/s/Bf7uVhGiu47uOELjmC5uXQ



Git Bash：Unix 与 Linux 风格的命令行，使用最多，推荐最多
Git CMD：Windows 风格的命令行
Git GUI：图形界面的 Git，不建议初学者使用，尽量先熟悉常用命令

1）cd：改变目录。

2）cd . .： 回退到上一个目录，直接 cd 进入默认目录

3）pwd：显示当前所在的目录路径。

4）ls(ll): 都是列出当前目录中的所有文件，只不过 ll(两个 ll)列出的内容更为详细。

5）touch：新建一个文件 如 touch index.js 就会在当前目录下新建一个 index.js 文件。

6）rm: 删除一个文件, rm index.js 就会把 index.js 文件删除。

7）mkdir: 新建一个目录,就是新建一个文件夹。

8）rm -r : 删除一个文件夹, rm -r src 删除 src 目录

rm -rf / 切勿在 Linux 中尝试！删除电脑中全部文件！

9）mv 移动文件, mv index.html src index.html 是我们要移动的文件, src 是目标文件夹,当然, 这样写,必须保证文件和目标文件夹在同一目录下。

10）reset 重新初始化终端/清屏。

11）clear 清屏。

12）history 查看命令历史。

13）help 帮助。

14）exit 退出。

15）#表示注释

16)git config -l 当前项目下 git 的配置

查看不同级别的配置文件：

#查看系统 config

git config --system --list

#查看当前用户（global）配置

git config --global --list

设置用户名和邮箱

当你安装 Git 后首先要做的事情是设置你的用户名称和 e-mail 地址。这是非常重要的，因为每次 Git 提交都会使用该信息。它被永远的嵌入到了你的提交中：建议 Git 的用户名和邮箱与 GitHub 的用户名和邮箱保持一致

git config --global user.name kuangshen #名称

git config --global user.email 24736743@qq.com #邮箱

Git 基本理论

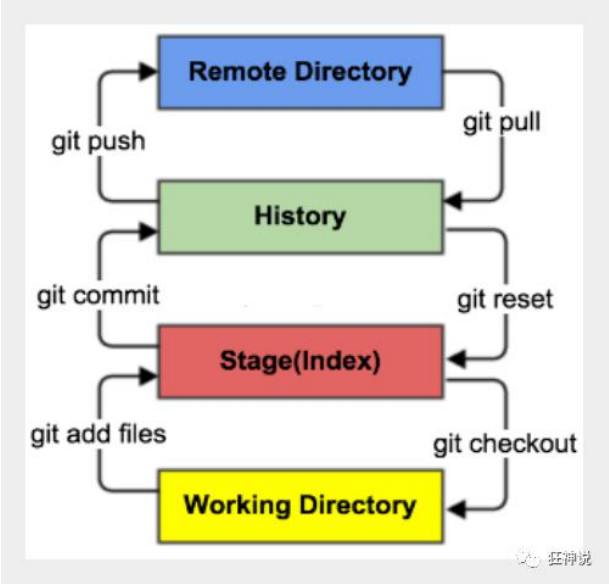
工作目录（Working Directory）、暂存区(Stage/Index)、资源库(Repository 或 Git Directory)。如果在加上远程的 git 仓库(Remote Directory)就可以分为四个工作区域。

git 的工作流程一般是这样的：

1、在工作目录中添加、修改文件；

2、将需要进行版本管理的文件放入暂存区域；

3、将暂存区域的文件提交到 git 仓库。

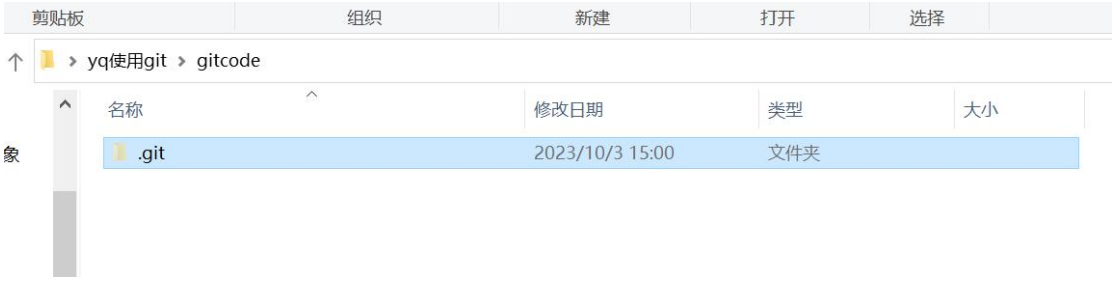


Git 项目的搭建

构建本地仓库的两种方法：

1 创建全新的仓库

git init #初始文件

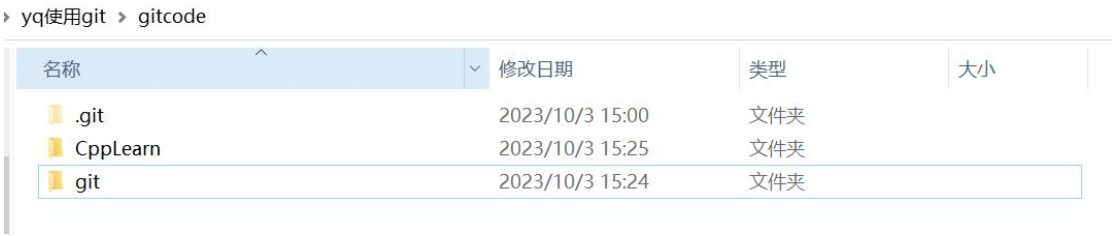


2. 克隆远程仓库

Git clone url

克隆一个项目和它的整个代码历史(版本信息)

```
yq@DESKTOP-AQH97H2 MINGW64 ~/Desktop/yq使用git/gitcode (master)
$ git clone https://github.com/Mrhelloyang/CppLearn.git
Cloning into 'CppLearn'...
remote: Enumerating objects: 145, done.
remote: Counting objects: 100% (86/86), done.
remote: Compressing objects: 100% (65/65), done.
remote: Total 145 (delta 32), reused 56 (delta 21), pack-reused 59
Receiving objects: 100% (145/145), 46.99 KiB | 528.00 KiB/s, done.
Resolving deltas: 100% (49/49), done.
```



Git 基本操作命令

#查看指定文件状态
git status 文件名 filename
#查看所有文件状态
git status
#添加所有文件到暂存区
git add .
#提交暂存区中的内容到本地仓库 -m 提交信息
git commit -m "消息内容"

忽略文件
有些时候我们不想把某些文件纳入版本控制中，比如数据库文件，临时文件，设计文件等
在主目录下建立".gitignore"文件，此文件有如下规则：

```
#为注释
*.txt          #忽略所有 .txt 结尾的文件,这样的话上传就不会被选中!
!lib.txt       #但 lib.txt 除外
/temp          #仅忽略项目根目录下的 TODO 文件,不包括其它目录 temp
build/         #忽略 build/目录下的所有文件
```

doc/*.txt　　#会忽略 doc/notes.txt 但不包括 doc/server/arch.txt，只忽略 doc 下一级的 txt 文件，不包括下下级的 txt 了

基于 SSH 协议配置 Git 连接 GitHub

这样就不用每次上传代码都输入密码，方便一点
首先检查下本机是否已经安装了 SSH，在终端输入 ssh 即可：

```
[guest@contos7 ~]$ ssh
usage: ssh [-1246AaCfGgKkMNnqsTtVvXxYy] [-b bind_address] [-c cipher_spec]
           [-D [bind_address:]port] [-E log_file] [-e escape_char]
           [-F configfile] [-I pkcs11] [-i identity_file]
           [-J [user@]host[:port]] [-L address] [-l login_name] [-m mac_spec]
           [-O ctl_cmd] [-o option] [-p port] [-Q query_option] [-R address]
           [-S ctl_path] [-W host:port] [-w local_tun[:remote_tun]]
           [user@]hostname [command]
[guest@contos7 ~]$ ll
```

如果没有安装进行 yum 安装：

yum -y install openssh-clients

ssh-keygen -b 4096 -t rsa -C "xxxx@163.com"//用 github 的邮箱

在当前用户的根目录下打开.ssh 文件夹就会看到两个文件，分别是：id_rsa.pub（公钥），id_rsa（私钥）。将 id_rsa.pub 里面复制到 github 里面去，title 可为空。