**版本控制工具git**

1. **版本控制工具的作用**

**场景一：一个团队开发一个项目，指定张三疯写了一个公共的函数库 tools.js，应该分发给每个成员。李思峰在使用时，发现tools.js里的 函数 fnA有问题，李思峰会怎么办？改自己机子上的tools.js。接着怎么办? 发给张三疯，张三疯在分发给每个人（就害怕，李思峰忘了）**

**场景二：李思峰和李武峰都都发现 函数 fnA有问题（但是问题又不一样），都会改，改了后怎么办？发给张三疯，以谁的为准………………**

**场景三：前一天写的代码没有问题，第二天，增加功能时，改了前一天的代码，发现改坏了，想回到前一天的代码，没法回去了？**

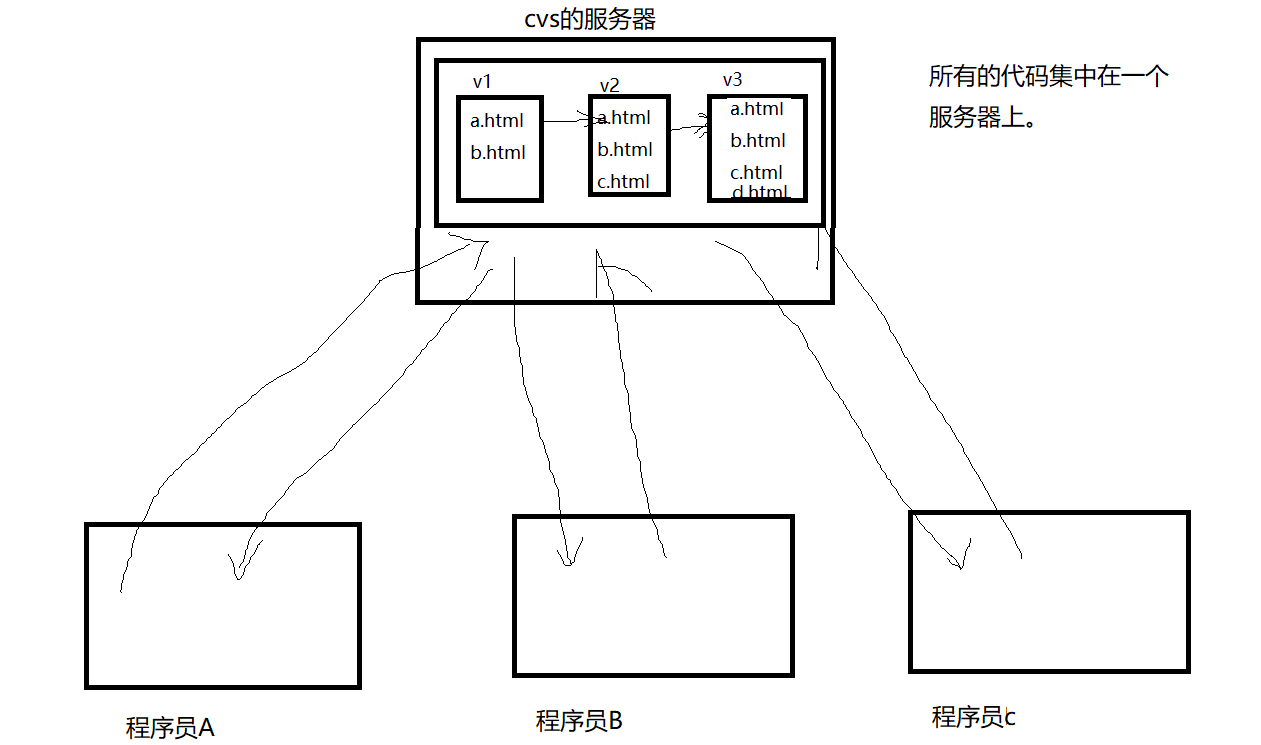
**…………………………………………**

**版本控制工具能够有效地解决以上问题。**

**版本控制工具解决了多人协作开发中的文件版本问题，方便回退，对比不同版本的区别，整个项目里程碑的版本等等。**

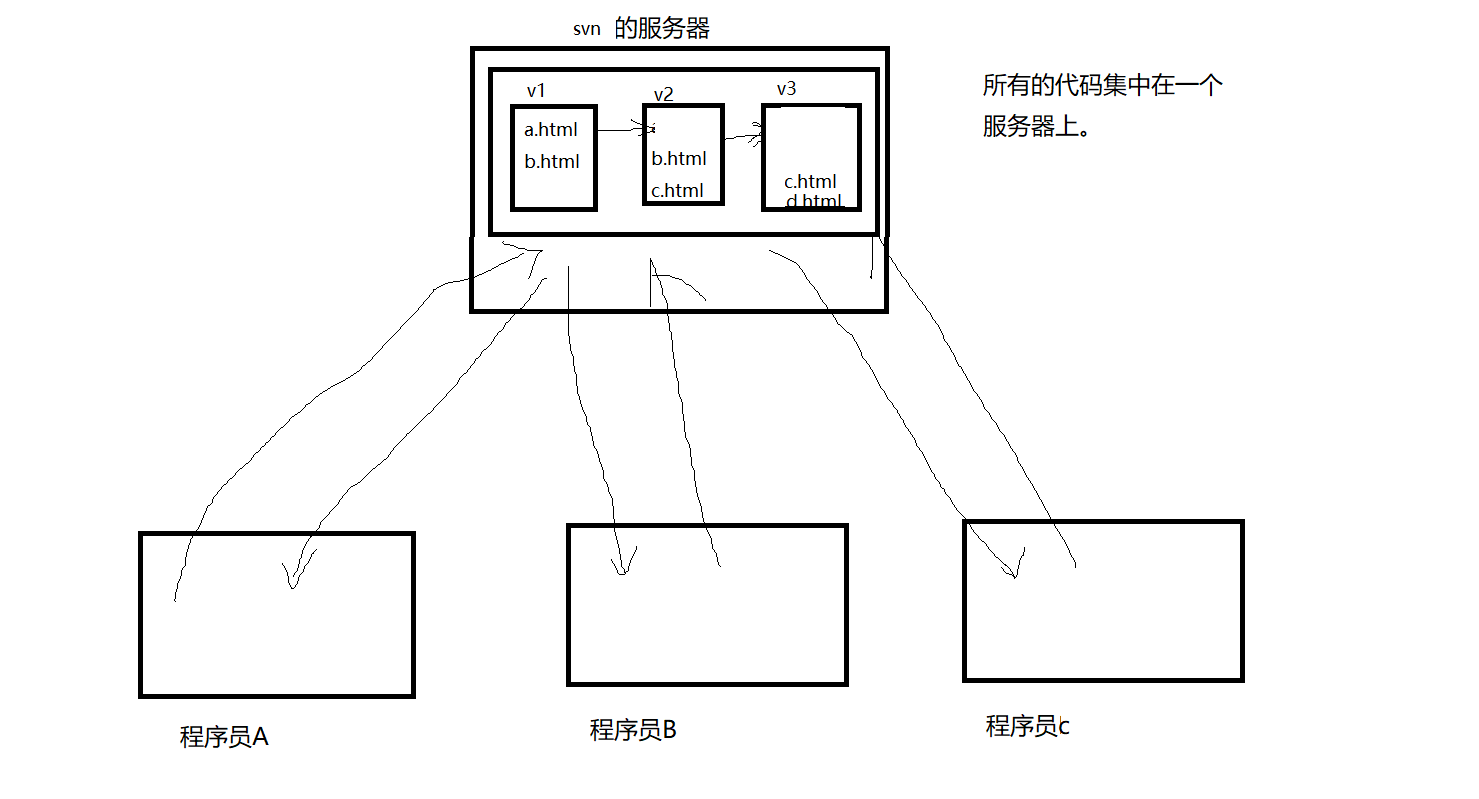
1. **版本控制工具经历的阶段**
   1. **cvs阶段**

**集中式的版本控制工具：**



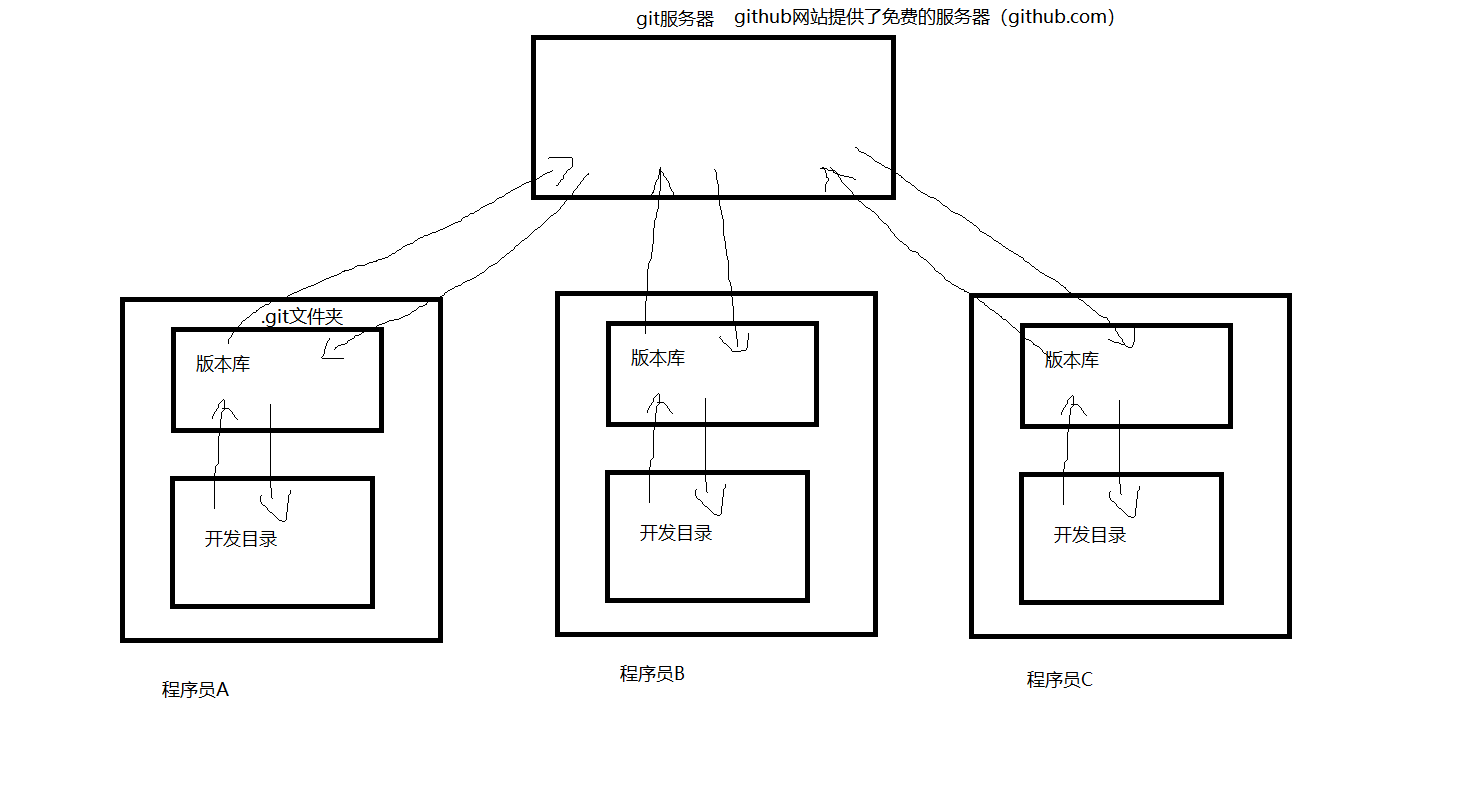
Cvs的做法：每次上传时，会把所有文件（不管改了没改）上传，即在每一个版本里，都是项目的所有文件，这样会造成大量硬盘空间的浪费，而且效率。这是缺点

* 1. **Svn阶段**为了解决cvs的问题，有了svn。Svn的特点是：每次上传的都是改过的，或者新加。这样效率高，也节约了服务器的空间。现在（2019年）依然有很多公司在使用svn。



* 1. **Git阶段**

Git是分布式版本管理工具，不管是cvs还是svn都是集中式的管理，有什么问题？集中式管理的问题？一旦服务器出问题了，版本管理就没法进行。



分布式的管理是：每个程序员对版本的管理都在自己的本地（就是自己的硬盘），这样对于版本的管理，就不再受服务器的影响。

git的特点：

1. 、Git对于不同版本内容的上传，不是以文件为单位。而是以代码行为单位，这样更加节约了服务器的空间。
2. 、git没有全局版本号。
3. 、git回退时，非常快。
4. **Git使用的准备工作：**
5. 安装
6. 初始化user.name,user.email
7. 进入git bash
8. 在命令行，输入以下内容（把your Name 改成你自己的用户名）：

git config --global user.name "Your Name"

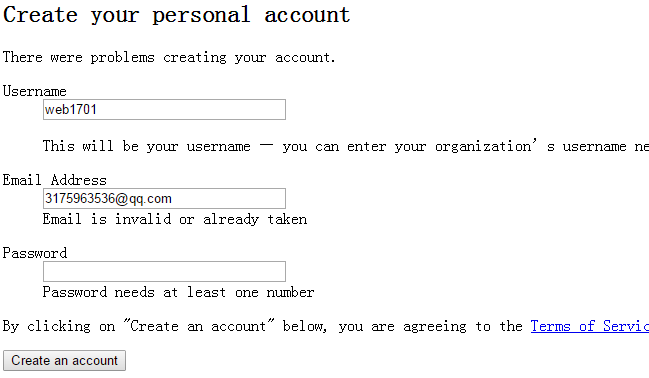
1. 在命令行，输入以下内容（把email@example.com 改成你自己的邮箱）：

git config --global user.email ["email@example.com"](mailto:\"email@example.com\")

1. **建立远程仓库**
   1. 在github.com网站上创建账户

**1）、进入github官网（<https://github.com/>）**

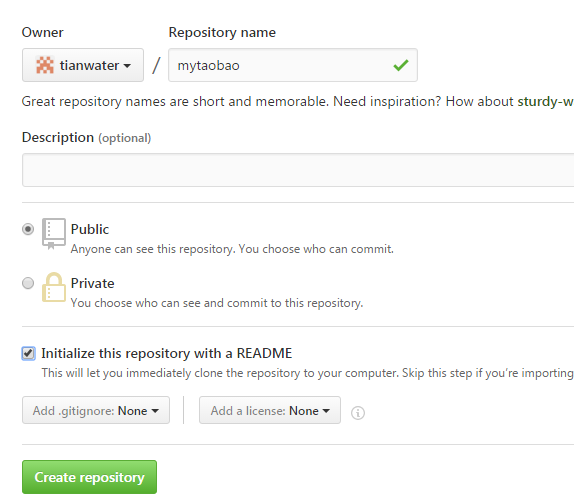
**2）、注册账号**

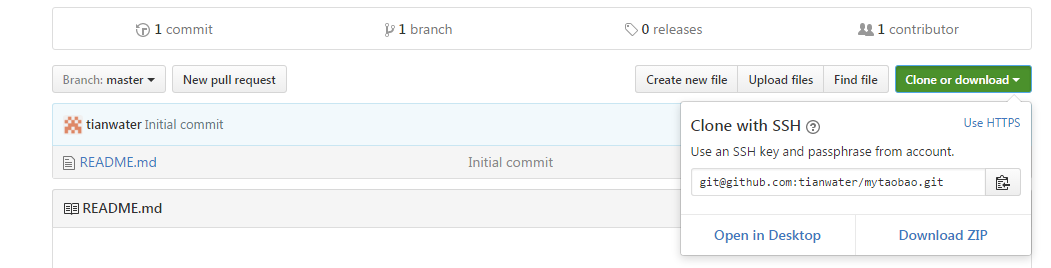


**3）、登录（Sign in）**



* 1. 建立远程仓库

1. 在页面中找 “new repository”，名字叫mytaobao：  
   

注意：勾选Initialize this repository with a README，这样GitHub会⾃动为我们创建⼀个 README.md⽂件。创建完毕后，可以看到README.md⽂件：****

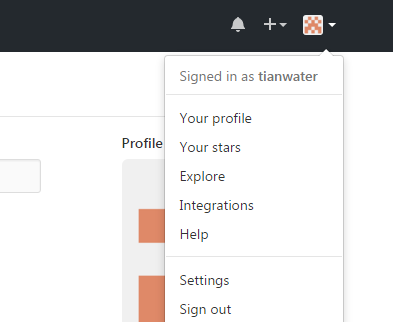
1. **建立本地仓库**
   * 1. 先建立本地和远程的连接（使用ssh）
     2. **在本地创建ssh key（公钥）**。

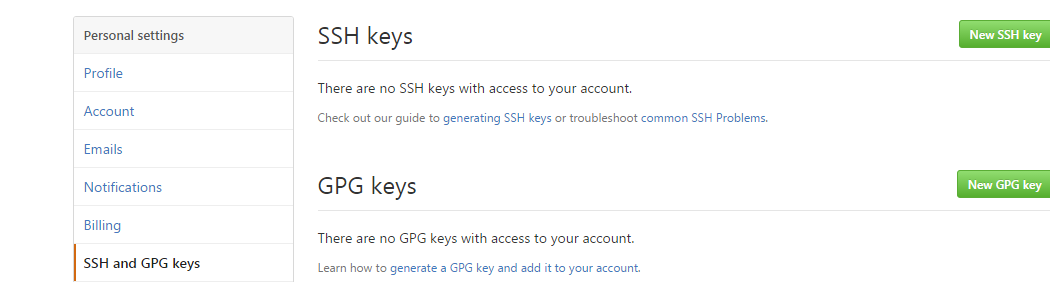
在git bash里输入： ssh-keygen -t rsa -C [youremail@example.com](mailto:youremail@example.com)你需要把邮件地址换成你⾃⼰的邮件地址，然后⼀路回车，使⽤默认值即可.

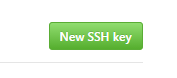
在⽤户主目录下(/c/Users/Administrator/.ssh/id\_rsa))，看看有没有.ssh目录，如果有，再看看这个目录下 有没有id\_rsa和id\_rsa.pub这两个⽂件，如果已经有了，可直接 跳到下⼀步。

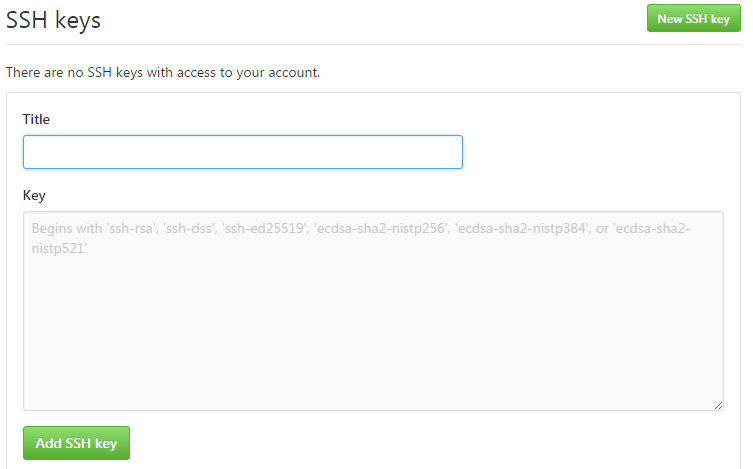
* + 1. **在远程创建ssh key：**

登陆GitHub，打开“Account settings”，“SSH Keys”⾯： 然后，点“Add SSH Key”，填上任意Title，在Key⽂本框⾥粘贴id\_rsa.pub⽂件的内容：









点击“add SSH Key”

* + 1. 再克隆远程仓库到本地

远程仓库地址：

git clone [git@github.com:tianwater/mytaobao.git](mailto:git@github.com:tianwater/mytaobao.git)

在你的当前文件夹下就会看到 mytaobao文件夹。

在此文件里有 .git 文件夹，这个文件夹就是本地版本库

1. **现在可以在项目中使用git进行版本管理了**
   1. 把开发的代码放在本地版本库
      1. Git add

假设在项目下有个index.html文件。在 git Bash，输入：git add index.html

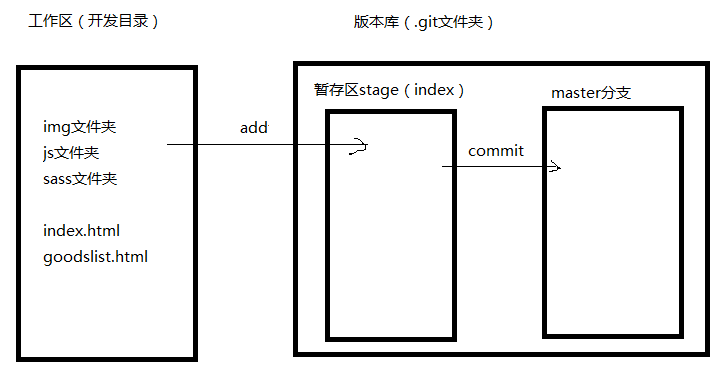
* + 1. Git commit(提交)

在git Bash里输入： git commit -m "这是新建的index.html"。其中"这是新建的index.html"只是说明或者说解释。

理解： add和commit；

Add:添加；相当于打了标记（实际上是存储在了暂存区stage里），告诉git，下次提交时把该文件进行提交。

Commit：提交。把所有打过标记的（即用add进行添加的文件），一次性进行提交（此时才有版本号）。即可以一次性把暂存区里的文件全部进行提交，提交了master分支（主分支）。提交完成后，暂存区里就没有文件了。



* 1. 恢复（跳转）到某个版本的文件

1. 、查看所有的版本的命令

git log

1. 、要恢复到上一个版本

git reset --hard HEAD^

1. 、恢复到指定版本

git reset --hard 版本序列号 （可以用版本号的前七位就行）

* 1. 把本地版本库的代码放在远程库

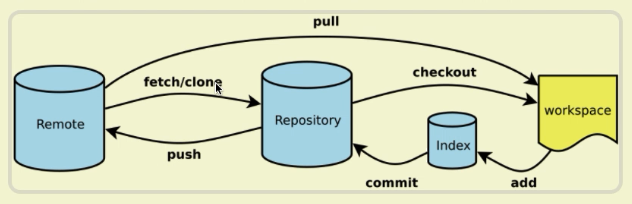
git push -u [git@github.com:tianwater/mytaobao.git](mailto:git@github.com:tianwater/bigMi.git) master

1. **总结：**

以后，每天做的git相关工作

1. 、git add 文件名
2. 、git commit –m “版本描述”

3）、git push -u 远端地址master



1. **补充：git和gulp应用于现有的项目**
   1. 远程建立仓库（如：taobao1809），git clone到本地，并自动创建项目文件夹(taobao1809)
   2. 在项目文件夹中(taobao1809)，npm init，并本地安装gulp和gulp插件
   3. 把现有的代码（html,css,图片文件等等）拷贝到项目目录(taobao1809)下
   4. 编辑.gitignore文件
   5. 把当前项目文件夹下的所有内容（不包括node \_modules文件夹等等）add到暂存区，并且commit到版本库